Object Recognition in Infrared Imagery Using Appearance-based Methods

Xun Wang

Thesis submitted for the degree of Doctor of Philosophy

Heriot-Watt University

School of Engineering and Physical Sciences

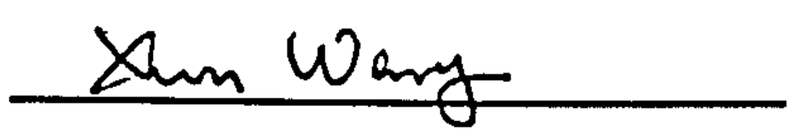
Department of Electrical, Electronic and Computing Engineering

<June> <2008>

Declaration

I hereby declare that the work presented in this thesis was carried out by myself at Heriot-Watt University, except where due acknowledgment is made, and has not been submitted for any other degree.

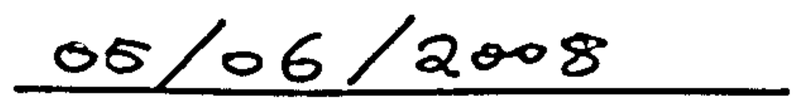
Author:



Xun Wang

Supervisor:

Prof. Andrew M. Wallace



Date

Abstract

Object recognition in infrared imagery has important applications, for example, in security and defence, and surveillance, due to the passive night-time and bad weather capabilities of infrared sensors. The objective of this thesis is to find a preferred method for the identification of static targets in single infrared images, concentrating on appearance-based methods. This has included thermal modelling of infrared signatures and the identification of images of different objects with variation in pose and thermal state.

This thesis reviews several popular approaches in object recognition in visible and infrared imagery, concentrating on the appearance based approach. Using principal component analysis, the variances among the images are extracted and represented in a low-dimensional feature Eigenspace. Any new image can be projected into the Eigenspace by taking an inner product with the basis. The object of interest can be recognized by a nearest-neighbour classification rule, made more accurate by application of over-sampling to the surface manifold by B-spline surface fitting, and made more efficient by a k-d tree search algorithm. To address the problems of recognizing targets in noisy and cluttered images, we have also employed a random sampling approach that is based on the principle of high-breakdown point estimation.

As a final step, a probabilistic framework in employed to improve the recognition rate and give a confidence measure for the result. The probability is determined by two facts: distance from the Eigenspace and distance in the Eigenspace. Using this probabilistic framework, we set an 'image window' on the test image and adjust the position of the window according to the recognition result in the form of probability. The 'image window' method makes the system able to bear small in-plane transformation of the object in the test image and to recognize poorly segmented test images.

We also discuss the possibility of using a non-linear dimensionality reduction method, Isomap, to replace PCA as the basis of data decomposition in the appearance based

method. Results show that although Isomap has some advantages in separating poses, it does not improve the recognition result using sufficient basis vectors compared to PCA. Therefore, we still use PCA as the basis for dimensionality reduction.

A new way of modelling thermal change has been proposed under the framework of an appearance based method. Possible thermal state changes of an object are modelled by several single component changes and the combinations of these changes. Hence, we build an Eigenspace model in which each object is represented by several lines (or vectors) in the Eigenspace and each line represents one pose and one thermal state change. Using this model, it is possible to predict subspace projection of changes in thermal state and to recognize new unseen thermal images. Using a recognition algorithm that measures scene to model similarity by the distance between the unknown point and the learnt linear object representation, we are able to show an improvement in the recognition accuracy over the conventional appearance based approach.

We have made extensive use of simulated data for both learning and recognising targets by appearance. As we have two degrees of freedom in viewpoint, azimuth and elevation, and several further degrees of freedom in allowing thermal state changes on different parts of the object, we have used as many as 33700 thermal images for a single object in the most extreme case. Hence, it is not feasible to both control the thermal state and acquire infrared data for the complete set of objects and viewpoints in the learning phase. In the recognition phase, we have used simulated data to test the algorithms, but have also embedded simulated vehicles within real infrared image data, a practice which is common in the literature on IR target recognition which is reviewed in Chapter 2. Although the simulation package, CameoSim, has been evaluated in comparison with real data, this is less than ideal, but necessary in the circumstances to evaluate and test the approach.

Acknowledgements

I would like to express my deep and sincere gratitude to my supervisor, Prof. Andrew Wallace, for his guidance, valuable comments, patience and support.

I would like to thank Prof. Emanuele Trucco for his support as my secondary supervisor. Thanks to Dr. Matt Kitchin for his support by providing IR image data for this research. In addition, I would like to thank the vision group members Angel, Arvind, Sergio and Zsolt for their valuable discussions and advice. Thanks to my friend Huiyu for his continuous support and encouragement.

I would like to thank BAE Systems and SELEX Sensors & Airborne Systems for their support under a research scholarship.

Finally, my thanks go to my parents and my husband for their support, love and encouragement.

ACADEMIC REGISTRY
Research Thesis Submission

Name:	XUN WANG		
School/PGI:	EPS/EECE		
Version: *(i.e. First, Resubmission, Final)*	FINAL	Degree Sought:	Ph.D.

Declaration

In accordance with the appropriate regulations I hereby submit my thesis and I declare that:

1) the thesis embodies the results of my own work and has been composed by myself
2) where appropriate, I have made acknowledgement of the work of others and have made reference to work carried out in collaboration with other persons
3) the thesis is the correct version of the thesis for submission and is the same version as any electronic versions submitted*.
4) my thesis for the award referred to, deposited in the Heriot-Watt University Library, should be made available for loan or photocopying and be available via the Institutional Repository, subject to such conditions as the Librarian may require
5) I understand that as a student of the University I am required to abide by the Regulations of the University and to conform to its discipline.

* *Please note that it is the responsibility of the candidate to ensure that the correct version of the thesis is submitted.*

Signature of Candidate:	Xun Wang	Date:	05/06/2008

Submission

Submitted By *(name in capitals)*:	XUN WANG M. CHANTLER.
Signature of Individual Submitting:	Xun Wang
Date Submitted:	6/6/2008.

For Completion in Academic Registry

Received in the Academic Registry by *(name in capitals)*:	Amrittal		
1.1 Method of Submission *(Handed in to Academic Registry; posted through internal/external mail)*:	handed in		
1.2 E-thesis Submitted			
Signature:		Date:	6/6/08

Table of Contents

Lists of Figures

Chapter 1

Introduction

1.1 Motivation

The aim of an object recognition system is to know 'what it is' in the scene. The oldest image forming and object recognition system is our visual system. With our eyes as the sensor, via optic nerves, our brain can identify the object in the scene. The human eye-brain system has been the ultimate objective in computer vision system design. However, the human visual system is not perfect. For example, the eye is limited in the wavelengths to which it is sensitive, it does not see well at night, and it does not have capability at extended range. In response to these limitations, humans have developed devices to view our environment far beyond our human sensing systems. The object recognition system here is to process all the information and make decisions based upon these data.

Infrared imagery is particularly of interest because in many applications, e.g., military, the imaging systems have to work day and night and during obscuring weather. The passive night-time and bad weather capabilities of infrared sensors are important. Another motivation for exploring infrared imaging in this context is that research in other areas has shown an improvement using infrared images over algorithms using visible images. This is due to the fact that the majority of light in the longer thermal infrared wavelengths is from direct thermal emission. Since there is relatively little reflected light, images are largely invariant to changes in lighting conditions.

A typical automatic target recognition (ATR) system may include several algorithmic components, such as pre-processing, target detection [1] [2] [3], segmentation, target recognition [4] [5] [6] [7], prioritisation, tracking [8] [9] [10] and aimpoint selection. The intention of this thesis is to concentrate primarily on the target recognition scenario, assuming there is sufficient pixel information to do more than just detect a target. We adopt the appearance-based method to solve this recognition problem because this approach is more reliable than extracting local features provided there are sufficient training images and test images of good quality. The proposed recognition system in this thesis is mainly to release those constraints of the original appearance-based method.

One problem of original appearance-based object recognition is that it requires a large set of training images from different camera positions and under different lighting conditions for successful recognition. In this research, we manage to achieve a comparative result using interpolation based fewer pose images. Further, in infrared imagery, the appearance of the object is affected by its thermal state and it is difficult to get images from all possible thermal states of an object. We propose a method to model changes in thermal states within the appearance-based framework. The method models a relatively small set of thermal states and can recognize images containing the same object with new thermal states. In the experiments, in order to control the thermal states and acquire infrared images from designed viewpoints, we use simulated infrared images generated by CameoSim in this thesis, which has been evaluated elsewhere in comparison with real data.

In recognition, the original appearance-based method requires a good segmentation of the test image. The object is supposed to be strictly in the centre of the test image and a shift of several pixels from the centre will affect the recognition result. In this thesis, we propose a method which is able to recognize images in which the target object may have more flexible positions. The proposed object recognition system can also identify an object in test images with noise and occlusion.

1.2 Contribution

This thesis addresses the problem of object recognition in infrared imagery using appearance-based methods. There are two main contributions of this thesis: first, I propose a method to model changes in thermal state using an Eigenspace; second, I propose a probabilistic framework to measure the confidence of the recognition result.

The original appearance-based method is designed for visible imagery. Although several researchers [11] [12] [13] have used an appearance-based method to deal with infrared images, none of them have addressed the problem of changes in thermal state. One primary difference between an infrared image and a visible image is that the state of inner component of an object could affect the appearance, e.g., as the engine heats up, the appearance of a vehicle changes. In this thesis, I model all possible thermal state changes of an object by several single component changes and the combinations of these single component changes. Taking a vehicle for example, the thermal state changes can be engine state changes, exhaust state changes, tyre state changes and a combination of all three. We show that when projecting the image of an object from a single pose with a single state change, e.g., as engine heats up, the projection of those images can approximate a line in a multidimensional space. We have proved this theoretically and empirically. Based on this finding, we build an Eigenspace model that accounts for thermal state changes. In the model, each object is represented by several lines (or vectors) in the Eigenspace and each line represents one pose and one thermal state change. The nice feature of this model is that it can predict subspace projection of thermal state changing and is able to recognize any new thermal images having an unknown thermal state.

The second main contribution is that we propose a method to assign to the recognition result a certain confidence, i.e., the probability of object A being present in the test image is higher than other objects. This is an expansion of the general appearance based method. Although Moghaddam and Pentland [14] used a probabilistic Eigenspace, their framework is only for face recognition. In this thesis, we propose a probabilistic framework for

general 3D object recognition. The probability is determined by two facts: distance from the Eigenspace and distance in the Eigenspace. The second of these has been used as the only criterion for object recognition in the original appearance-based method. We are able to demonstrate that both are important and establish formulas to calculate them in this thesis. Using this probabilistic framework, we set an 'image window' on the test image and adjust the position of the window according to the recognition result in the form of probability (see section 3.5.4). The 'image window' method makes the system able to cope with small in-plane transformations of the object in the test image and to recognize objects in not well segmented test images.

1.3 Thesis Structure

In Chapter 2, we review approaches in general and appearance-based object recognition in detail including the most original research and current developments to deal with problems of occlusion, runtime segmentation and ambiguity between objects. The second part of Chapter 2 reviews research on object recognition in infrared imagery in which the thermophysical invariant approach is discussed in detail.

In Chapter 3, we describe the theory of the basic appearance based object recognition method including the processes of training and recognition, the relation between the distance measurement in Eigenspace and the correlation between images, and two ways of building the Eigenspace, the Universal Eigenspace and Object Eigenspace. To improve the accuracy and efficiency of the algorithm, we implement the methods of interpolation of the manifold and k-d tree searching. To solve the problem caused by noise and occlusion in test image, we implement a robust sampling method adopted from Leonardis and Bischof's work [15] [16]. We describe the theory of this method and demonstrate the improvements through experiments. Finally, we propose a probabilistic framework, in which we explore the theory of the original Eigenspace based method, and extend the single measurement of the original method, distance in Eigenspace, to two by adding the

distance from Eigenspace as another important factor. We describe how to measure those two factors and the theory and implementation of the probabilistic framework. We have tested these algorithms on both visible and infrared images.

In Chapter 5, we describe our proposed method to model thermal state change. The core of the method is to predict the projection of an object with a new thermal state in Eigenspace. This includes both single-part thermal state change and multi-part thermal state change. The proposed prediction is proved theoretically. Experiments have been done to compare the proposed method with the original method.

We have also examined a nonlinear subspace method – Isomap as a candidate to replace PCA as the basis for appearance based object recognition in Chapter 4. We compare the performance of this nonlinear method with PCA and discuss the advantages and disadvantages.

Chapter 2

Literature Review

2.1 Overview of Object Recognition

The problem in object recognition is to determine which of a given set of objects appears in a given input image. A general object recognition scheme begins by building the target templates from models of the targets and then matching them to target features from real images to fulfil the recognition task [17].

Figure 2-1 shows the two stages in object recognition systems: Modelling and Recognition. In the first stage, given a collection of named objects, a model library is constructed from certain descriptions of the objects. In the second stage, given a sensor image, the system is to determine the identity of any library objects in the image and sometimes the orientation of the object. In general, recognition is the process of finding a correspondence between certain features in the image and similar features of the object model. The most important issues involved in the process are: (a) identifying the type of features to use and (b) determining the best procedure to establish the correspondence between image and model features. These two issues are referred to as the Modelling and Matching process as shown in Figure 2-1. The reliability and efficiency of an object recognition system directly depends on how carefully these issues are addressed. Although the *Image Formation* and *Feature Extraction* are two important processes in the whole object recognition system, they are often regarded as low-level or middle-level

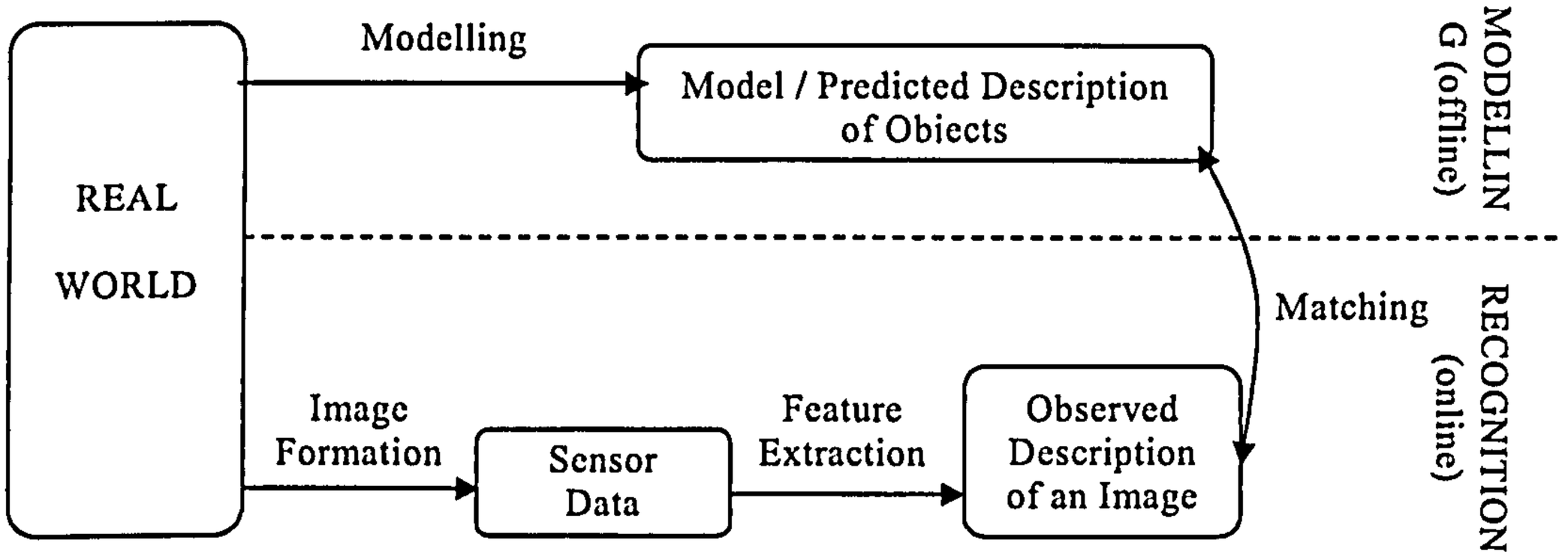


Figure 2-1 Framework of a commonly used object recognition scheme

image processing. In this chapter, we review some main approaches of object recognition algorithms, concentrating on their *Modelling* and *Matching* process.

2.1.1 Modelling by invariants

One difficulty in 3D object recognition is that the object view will be subject to certain transformations: affine transformation and perspective transformations depending on the viewpoint, transformations related to the lighting conditions and other possible factors.

Early researches on object recognition are concentrated on modelling objects under affine transformations [18] [19] [20]. In Lamdan's work [18], they extract so-called interesting points both in the object model images in the scene image to find the best match between those point sets using Hough-based method. Among these interesting points, each non-collinear triplet of points forms a 2D linear basis. One can express the coordinates of all other model points in this basis. All the basis and coordinate of the model points are stored in a hash table.

Recognition is based on the fact that any affine transformation applied to the set of points will not change the set of coordinates based on the same ordered basis triple (Figure 2-2). In the recognition stage, given an image, they extract the interesting points, choose a

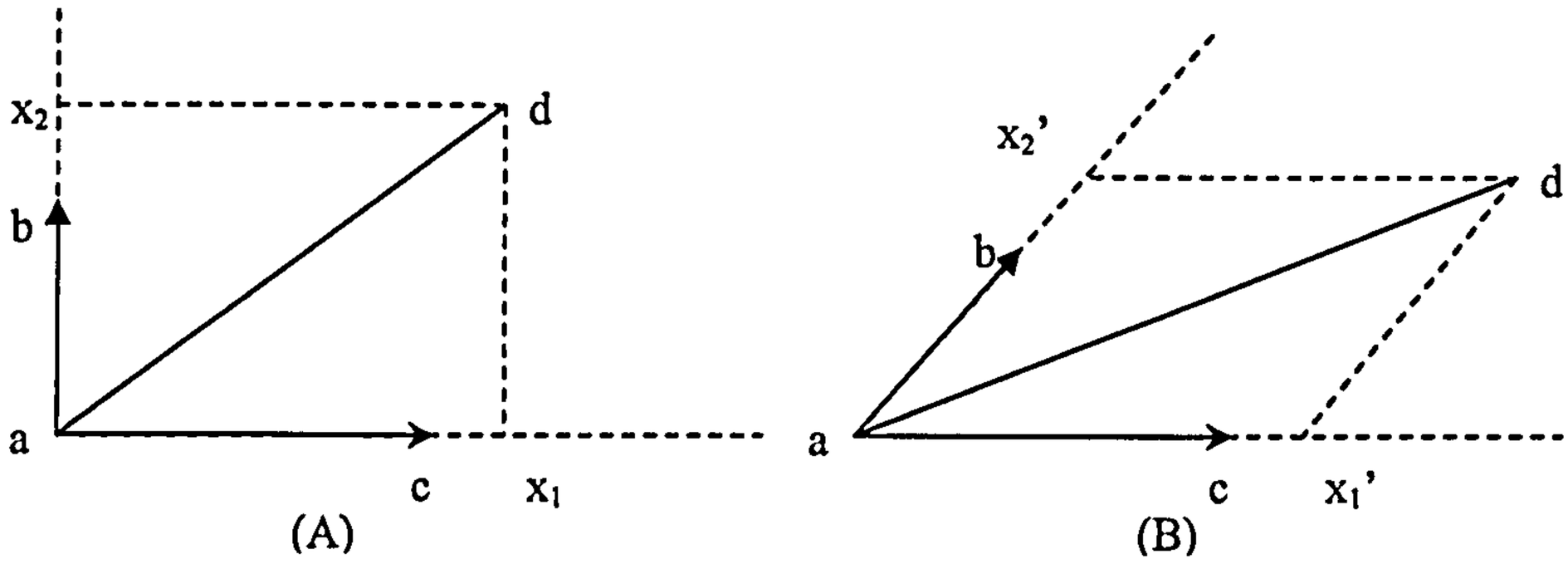


Figure 2-2 Coordinates of point d in the affine basis triplet (a, b, c). The coordinates of point d before (x_1, x_2) and after (x_1', x_2') the affine transformation are identical.

triplet of non-collinear points as a basis triple, and compute the coordinate of the other points in this scene. For each such coordinate, they check the appropriate entry in the hash table, find the pairs (model, basis triplet) that appear, and give a vote for the corresponding model. They then find the pairs that obtained a large number of votes. If there is no high scoring pair, they continue by checking another basis triplet in the image. After several cycles of choosing triplet of points and matching, the object of interest is finally identified.

Arbter et al. [19] modelled the objects by their shape using features which are affine-invariant. Their method is based on a parameterized boundary description, which is transformed to the Fourier domain and normalized there to eliminate dependence on affine transformation (translation, rotation, scaling and shearing). The use of Fourier Descriptors as a representation for closed curves was firstly suggested by Cosgriff [21]. Initially a curve is plotted as tangential orientation against arc length. The resulting one dimensional boundary profile is then normalized to a length of 2 and then expanded as Fourier series using the Fourier expansion. The boundary is then uniquely represented by the infinite series of Fourier coefficients. In practice this series can be shortened to give a finite shape descriptor whilst still retaining sufficient descriptive power. Thus the closed curve can be represented by a periodic functions of a continuous parameter, or the set of Fourier coefficients of this function. These Fourier coefficients are referred to as 'Fourier descriptors'. Realizing the traditional parameterization, arc length, is nonlinearly

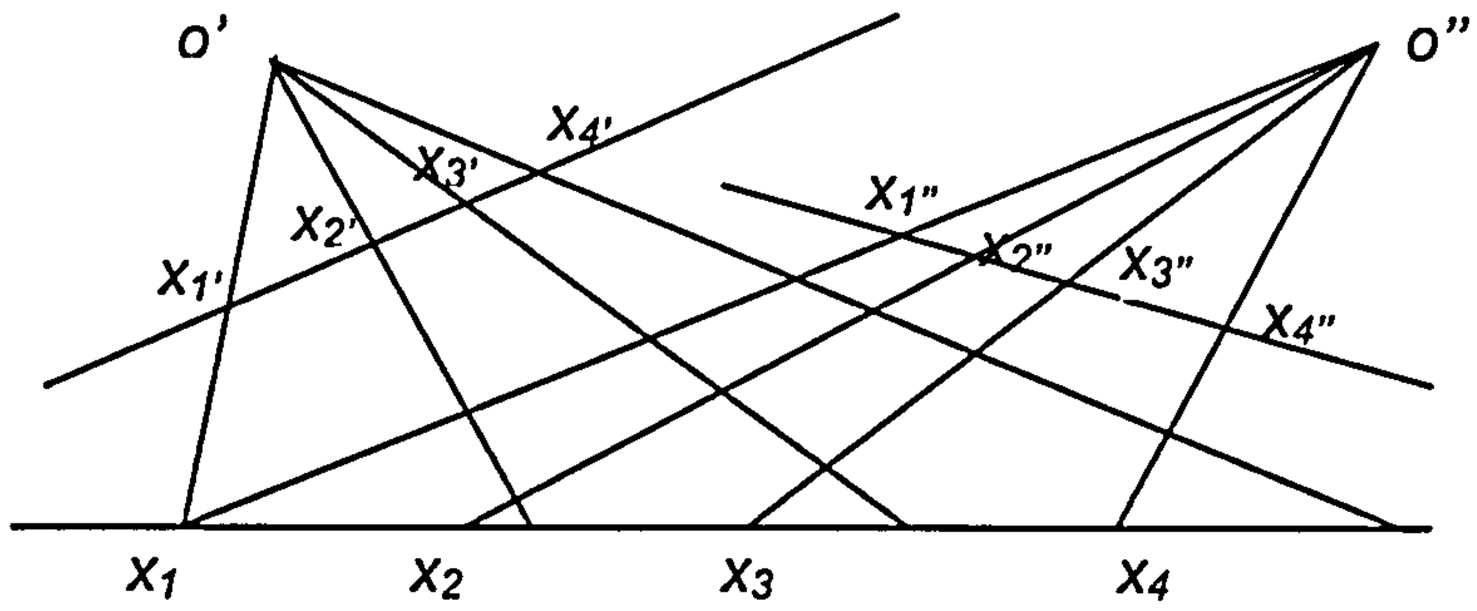


Figure 2-3 Projection geometry of straight lines

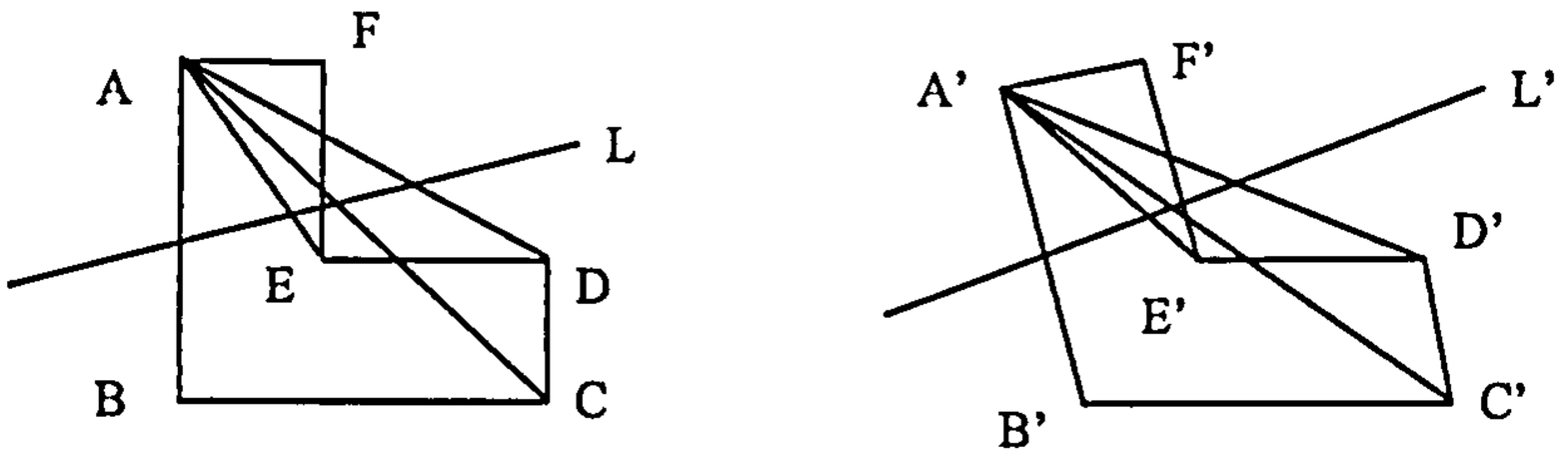


Figure 2-4 Left: Spatial Planar , Right: Projective Planar.

transformed under an affine transformation, Arbter et al. define a new parameterization based on the idea of 'affine length'. The new parameterization is linear under affine transformation and independent of the initial representation of the contour.

In the above methods, affine invariants were used to recognize planar objects in 3D space. Orthographic projection was used to approximate perspective projection. Hence, the assumption is that the size of the observed object is small relative to its distance from the camera.

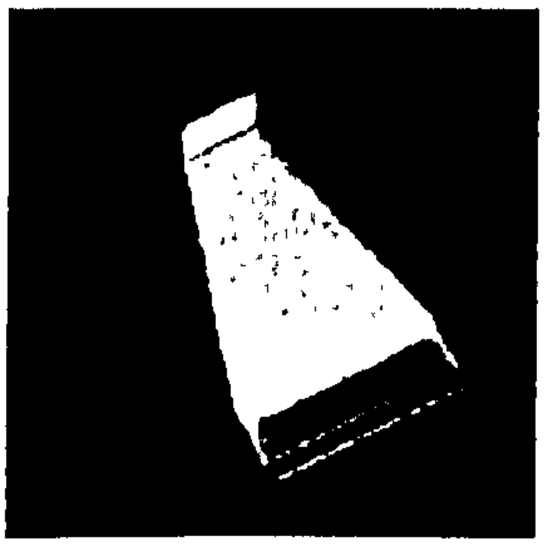
Under large perspective effects, perspective invariants are needed. The perspective transformation consists of Euclidean motion in space composed with perspective projection. The perspective projection involves more difficult mathematics than does the parallel one. An example is that the parallelism of lines is invariant under parallel projection but not under perspective projection. Lei [22] demonstrated how cross ratios can be used to recognize a planar object in 3D space. As a reminder, the cross ratio of four collinear points is defined as $CR(x_1,x_2,x_3,x_4)=(x_1-x_4)(x_2-x_3)/(x_1-x_3)(x_2-x_4)$ where x_1,x_2,x_3,x_4 represent the

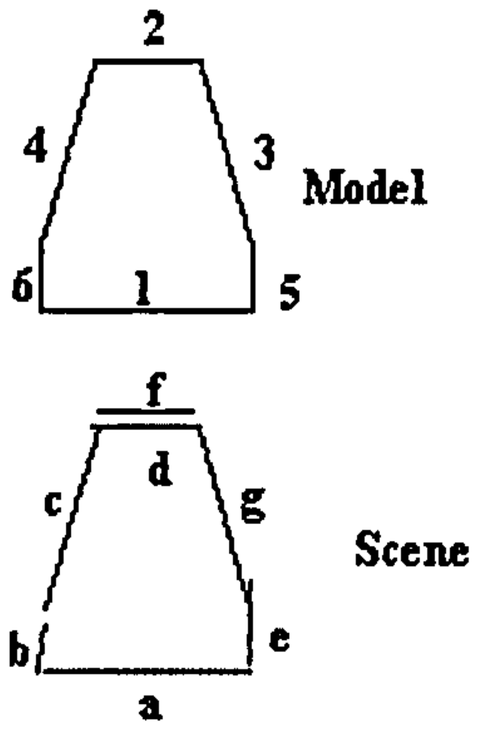
corresponding positions of each point along the line. It is known that the cross-ratio is an invariant of any sets of four collinear points in projective correspondence. For example, in the projection geometry shown in Figure 2-3, $CR(x_1, x_2, x_3, x_4) = CR(x_1', x_2', x_3', x_4') = CR(x_1'', x_2'', x_3'', x_4'')$.

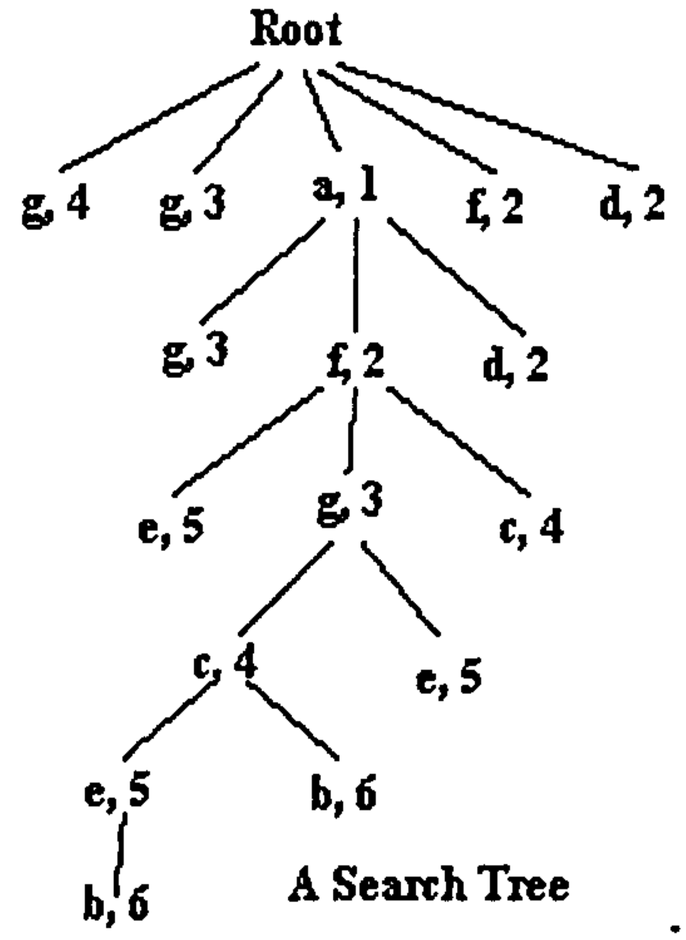
In Lei's work, he used CR as perspective invariant shape descriptor. The CR in his work is calculated using the four projection lines formed by drawing four line segments on the spatial plane of the polygon from one vertex to other four successive vertices. They proved that the CR remains constant in the projective plane. Thus, the CR reflects the vertex structure of shape of the polygon. They calculate the CRs of each vertex to obtain a sequence of CRs of the planar polygon and use this sequence as the shape descriptor and the feature for recognition. The minimum mean square error is used as the matching criterion. However, in this work, objects were restricted to polygons and required accurate identification of vertex positions. Forsyth et al. [23] constructed shape descriptors that are invariant under projective transformation and could represent curve features in the object.

Illumination invariants have also been studied extensively. These allow for changes that may include the altering of the position and number of light sources, the brightness. Swain and Ballard [24] introduced the concept of colour histograms for indexing objects in image databases, proving that colour can be exploited as a useful feature for rapid detection. They demonstrate that colour distributions without geometric information could be used to recognize objects efficiently from a large database of models. However, the performance degrades significantly when the illumination cannot be controlled. Slater and Healey [25] used local colour invariants to recognize 3D objects. In their study, they derive invariants of local colour pixel distributions, which were independent of the position and orientation of an object's surface. Following recognition, geometric information was used to estimate object location and orientation.

The invariant based approaches described above rely on good pre-processing of the input image, e.g., extraction of low-level cues such as corners, edges, local shading and texture. This is the challenge of this approach when applied to recognize complex objects: the







(a) (b) (c)

Figure 2-5 (a)A picture of the object -- a printer bracket; (b) Symbolic representation of the model and the scene; (c)the search tree. Source from [26]

feature extraction procedure is not stable under various imaging conditions, e.g., noise, occlusion, varying background, etc..

2.1.2 Matching Models to Scene Descriptions

Given an image and an object model, both represented in terms of their symbolic features, we want to find a partial match between the two. Obviously, a model feature can correspond to an image feature only if their properties are similar enough. This correspondence between model and image symbolic feature descriptors can be framed as a search in interpretation trees [27] [28] [29] [30]. The name refers to a search tree of choices concerning the interpretation of each image feature. From the root of the tree the match search examines an additional image feature at each level of the tree (see Figure 2-5).

The simplest approach is to search every leaf, which would be computationally very expensive. In order to avoid match hypotheses that are locally inconsistent, the search can incorporate two matching constraints: unary constrains and binary constraints. Grimson and Lozano-Perez [31] built a recognition system to recognize complex planar parts under

translation and rotation. They suggested the use of geometric constraints between image features to test feasibility and prune large portion of tree. They are unary constraints that apply to single features, e.g., size, and binary constraints between pairs of features, e.g., angle and distance. The two main drawbacks of the interpretation trees are its exponential complexity in the number of features, and the linear cost with respect to the number of models in the model base. Although the interpretation trees can be used on its own to do the matching, it is often combined with other matching methods to help to prune many of the possible matches between object features and image features prior to the more expensive matching steps, e.g., hypothesis and testing [26] or pose clustering [32].

The object could be represented not only by its features but also by the relations among features. The relation among features may by spatial or some other type. An object in such cases is represented as a graph consisting of multiple nodes, with each node representing a salient feature and each graph edges (the line connecting nodes) represent relations among the features in nodes. The object recognition problem is then considered as a graph matching problem [33] [34]. In this framework, the task is to find a common subgraph isomorphism between two attributed graphs: one representing the image and the other representing the model. Usually an inexact match is sought, where the attributes of matching nodes and edges are allowed to differ somewhat to accommodate distortion in the image. The search is guided by some measure of graph match quality, which evaluates both how well the two graph's structures match and how well their corresponding attribute values match.

The above two methods are using direct matching of the model to the unknown object and select the best-matching model to classify the object. These approaches consider each model in sequence and fit the model to image data to determine the similarity of the model to the image component. In other applications where the features in the images and the models can be normalized so that they can be presented in the same metric space, we can use classification method to do the matching. In these approaches, the features for the object can be represented as a point in multi-dimensional space. Examples of the

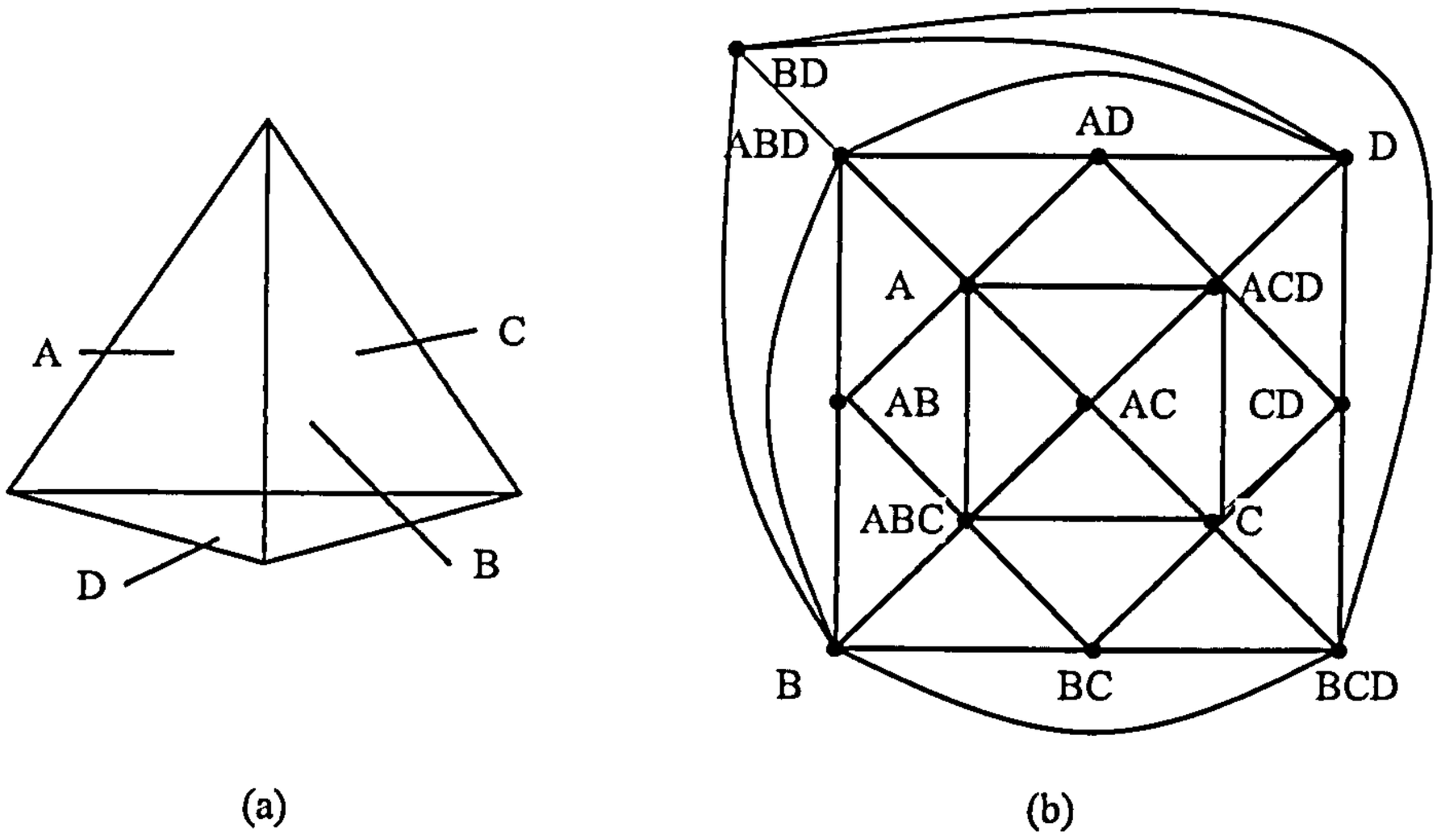


Figure 2-6 The aspect graph (b) of a tetrahedron (a). In (b), e.g., ABD represents the view from which face A, B and D are visible.

classification methods are Nearest Neighbour Classifiers, Bayesian Classifiers, Neural Nets, etc..

2.1.3 View-centred Representation and Recognition

Another approach to recognize 3D objects by computer is to obtain a series of 2D views of a known object, maintain them in some convenient representation in storage, and then match them against the views of an unknown object, thereby reducing the 3D problem to a series of 2D ones.

The aspect graph [35] [36] is a data structure that stores feature based information of views of objects (see Figure 2-6). The nodes of this graph represent the collection of stable views of the object which are the only significant views of that object. The aspect graph also maintains view adjacency information. Two stable views are said to be adjacent when it is possible to move a viewer to pass from one view to another with no intervening stable views. In the graph, each pair of adjacent stable views is joined by an edge. A popular algorithm to compute the aspect graph of an object is to tessellate an imaginary sphere which surrounds the object and compute the view from each vertex. These views can then be grouped into topologically similar views.

Aspect graphs have been defined for polyhedron[37][38], solids of revolution[39] and curved objects[40]. The main drawback of this approach is in the complexity of generating the aspects and in the large number of aspects which requires large storage and long search even for objects with modest complexity. Eggert et al. [41] observed that the aspect graph is often based on a level of detail not fully observable in practice. They explored a notion of a scale-space aspect graph to reduce the number of views. Ikeuchi and Kanade [42] use the similarity of feature extracted from 2D views of the object to form a graph structure used in recognition.

Dickinson et al.[43] constructed a hierarchical aspect graph system based on a set of primitives (parts). The difference is that traditional aspect graph representations of 3D objects model an entire object with a set of aspects, while their approach use aspects to represent a set of volumetric primitives from which each object in the database is constructed. Recognition is performed by first trying to recover the different 3D primitives in the image by comparing the image features with the aspect representations of the primitives. The recovered 3D primitives are then used to index the most likely complete 3D object.

2.1.4 Bag of Words methods

This class of methods for object categorization is motivated by an analogy to learning methods using the bag of words representation for text categorization [44] [45] [46] [47]. The key point of these methods is to build a codebook of commonly-accepted 'keywords' for images which is analogous to the dictionary of keywords for text documents. An image can then be represented by a sequence of these 'keywords', just as a text document is summarized as a sequence of keywords. In other words, the images can be represented by a histogram of the number of occurrences of particular image patterns. The discrimination of these image patterns makes them play a similar role like 'key words' in text. Early use of bag of words models in visual applications included texture recognition [48] [49] [50] [51].

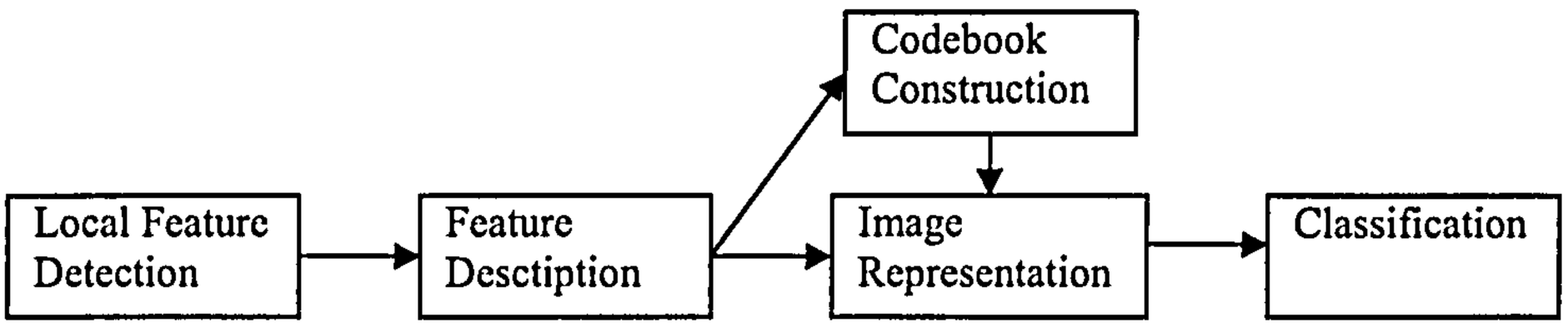


Figure 2-7 System structure of Bag of Words methods

Although the techniques vary, the main components of these methods are (see Figure 2-7):

- Local feature detection and description: These features are the basic elements of these methods. Each image, either a training image or a new input image, is firstly represented by a set of those feature descriptors.

- Codebook construction: The codebook, also called vocabularies in some context [52], is a set of cluster centres. The local feature descriptors for the first step are then assigned to the set of clusters. Depending on the way of local feature representation, these cluster centres are called keyblocks [53], or keypoints [52]. We'll call it keypoints in the following text.

- Image Representation: After the second step, each local image feature from the first step is associated with a cluster centre (keypoints). In this step, the method counts the number of local features assigned to each cluster. Each image is thus represented as a histogram for those keypoints, analogous to the representation of a text document as a list of keywords. This representation is called the bag of keypoints.

- Multi-class classification uses bag of keypoints as feature vectors, and thus determine to which category to assign to the image.

We now look into the details of these four steps.

Local feature detection and description:

The local features have been shown to be very powerful cues compared to the global features as they are more robust to occlusions and spatial variations [54]. In Zhu's work

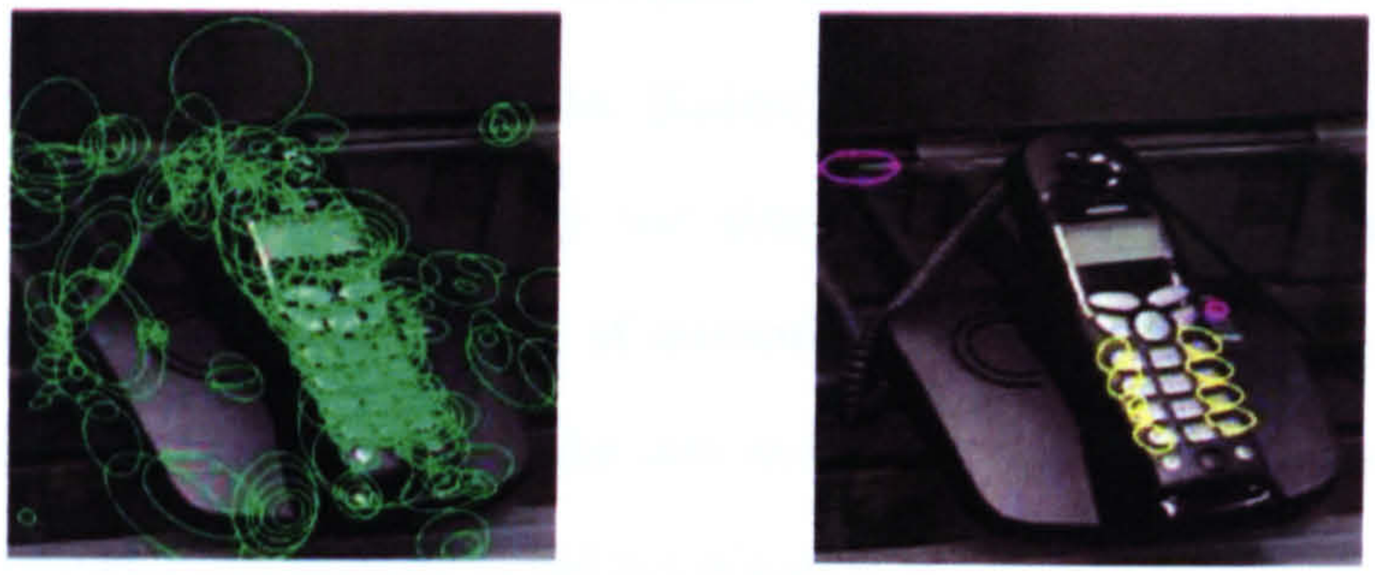
[53], images are partitioned into smaller blocks. Those blocks are the initial local features. In their experiment, they also examine different sizes of the blocks, e.g., 2x2, 4x4 and 8x8, and find that for image reconstruction, smaller block size yields lower distortion.

Csurka etc. [52] use a Harris affine detector [55] to detect affine invariant points and Scale Invariant Feature Transform (SIFT) descriptors [56] to describe the detected region. First, positions and scales of interest points are determined as local maxima of a scale-adapted Harris function. Then an elliptical neighbourhood is determined. The affine region is then mapped to a circular region to normalize it for affine transformation. Finally the image region is represented by multi-images using the SIFT descriptors to compute the Gaussian derivatives at 8 orientation planes.

Codebook construction:

The codebook contains codes for classification that relate new descriptors in query images to descriptors previously seen in training. Given the collection of detected local features from the training images, the codebook is constructed using a clustering algorithm. Csurka [52] and Li [57] and Sivic [58] uses a simple square-error partitioning method, k-means. This algorithm proceeds by iterated assignments of points to their closest cluster centres and recomputation of the cluster centres. To determine the number k, they run k-means several times with a different number of desired representative vectors k and different sets of initial cluster centres and select the final clustering giving the lowest empirical risk in categorization.

The cluster centres do not necessarily have a repeatable meaning such as 'car wheels' or 'nose'. However, in some cases when 'proper' cluster centres are selected, it could represent different groups of local features to some extent (see Figure 2-8).



(a) (b)

Figure 2-8 (a) All local patches detected (b) patches from two selected clusters occurring in this image (yellow and magenta ellipses). Source from [52]

Multi-class classification

The generative models are often used in multi-class classification, e.g., the Naïve Bayes classifier and the Hierarchical Bayesian text models, while some discriminative models, e.g., the SVM classifier finds a hyperplane which separates two-class data with maximal margin [59], and can be applied to multi-class problems by taking the one against-all approach to identify each class.

The Naïve Bayes classifier is often used in text categorization. It can be viewed as the maximum a posteriori probability classifier for a generative model in which each word in the document is chosen independently from a multinomial distribution over words specific to that class. For a set of labelled images $I = \{I_i\}$ and a codebook $V = \{v_i\}$ of representative keypoints, with each local feature labelled with the closest keypoints, the method counts the number $N(t,i)$ of the times keypoint v_i occurs in image I_i. To categorize a new input image, Bayes's rule is applied:

$$P(C_j \mid I_i) \propto P(C_j)P(I_i \mid C_j) = P(C_j)\prod_{t=1}^{|V|} P\left(v_t \mid C_j\right)^{N(t,i)} \tag{2-1}$$

In order to avoid probabilities of zero, the estimates are computed with Laplacian smoothing:

$$P(v_t \mid C_j) = \frac{1+\sum_{\{I_i \in C_j\}} N(t,i)}{|V|+\sum_{s=1}^{|V|}\sum_{\{I_i \in C_j\}} N(s,i)} \tag{2-2}$$

Further approaches of this class of methods include using pLSA (probabilistic Latent Semantic Analysis) [60] [58] or LDA (Latent Dirichlet Allocation) [61] [57] for the classification to handle the polysemy and synonymy of the 'words', and counting for special information [62]. These types of methods represent an object by histogram of its significant features. However, when the test images do not contain the significant image features of an object, this method could not discover it.

2.2 Appearance-based recognition – Early Research

Appearance-based models represent an object though a set of images. In contrast to bag of words methods, these images can be represented by a set of automatically derived 'feature' vectors by principal component analysis. Algorithms based on this model representation are called appearance-based recognition or Eigenspace-based recognition in this context, although appearance-based recognition also refers to e.g. aspect graphs or silhouettes in other literature [64].

2.2.1 The Application of Karhunen-Loeve Expansion / PCA to Face Recognition

Each image can be thought of as a point in a very high dimensional space in which the dimensionality is the number of pixels in the image, e.g., a 200x200 image is a point in a 40000-dimensional space. We call it image space. Although the appearance space of an object lies in the very high dimensional image space, it only actually occupies a much lower dimensional subspace of image space. One way of reducing the dimensionality of the appearance space of an object is to apply the Karhunen-Loeve expansion to the set of possible images of an object. Eigenspace-based object recognition has its origins in the work of Sirovich and Kirby [65] [66]. They applied the Karhunen-Loeve expansion to a set of training images, specifically human faces, in order to produce a low-dimensional description of faces. In other words, they determined the best coordinate system (in term of compression) for their training data.

The Karhunen-Loeve (K-L) expansion is a method for the compression of the information in a set of continuous functions into fewer and more important variables. When a set of functions is to be expressed as a series in terms of orthogonal and normalised base functions, the K-L expansion minimises the average error induced by taking only a finite number of these functions. In the discrete case the functions are replaced by vectors and the K-L expansion minimises the average error induced by using only a subset of the total set of orthonormal vectors needed to reconstruct a set of vectors. K-L expansion is often referred to as principal component analysis (PCA). The basis vectors for the new coordinate system are termed principal components, eigenpictures or eigenvectors.

Sirovich and Kirby demonstrated that any particular face can be economically represented in terms of a best coordinate system that they termed eigenpictures. First, a set of eigenimages is learned and then face images, including those not represented in the training set, were compressed by projecting them into the Eigenspace and storing the weights. Using the eigenimages and the weights, face images could be reconstructed. Here the reconstructed image may be an exact copy of the original image if all the eigenvectors are used. If however only a smaller set of those with the largest eigenvalues are used, the reconstructed image will be the best approximation to the original image for any basis with that number of dimensions. They derived the eigenpictures for a set of 115 face images and showed that the Eigenspace defined by the first 40 eigenvectors was sufficient in their case to reconstruct the training set to within a 3 percent error.

Realizing the technique's potential for speed, simplicity and learning capacity, Turk and Pentland [67] built the first recognition system based on Eigenspaces. They noted that in using an Eigenspace-based recognition strategy a small set of eigenpictures could be used to describe and reconstruct faces from a large segment of the population. The system is initialised in the following manner: a set of face images is acquired, the face images are normalised and aligned, the eigenvectors of this set are calculated and a certain number of those with the highest eigenvalues are retained. The weights for each individual are calculated by projecting back to the space. The weight vector for each person includes the

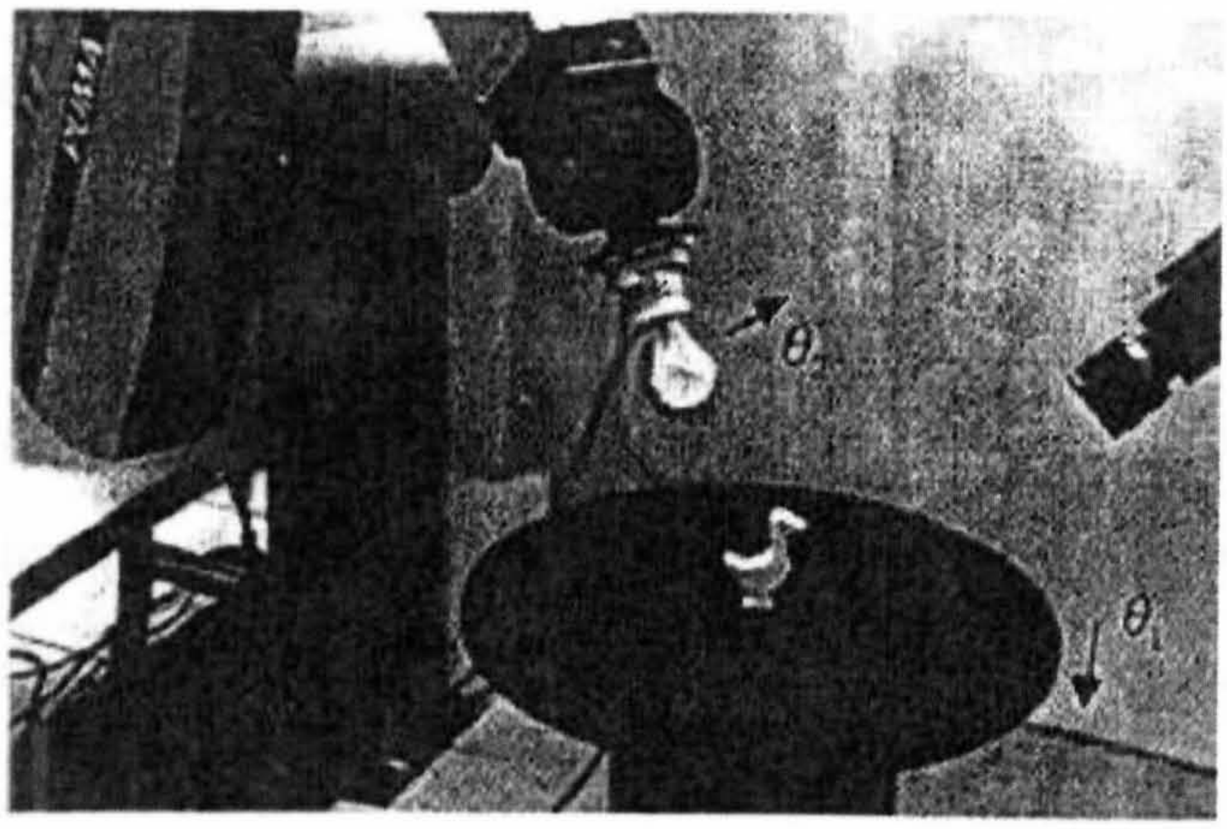


Figure 2-9 Setup used to automatically acquire image sets in Murase and Nayar's work [69]. The object is placed on a motorized turntable. Source from [69]

average weights calculated for that person over several images with slight changes in facial expressions, pose and lighting. In their experiments, 1 to 4 images for each individual were used.

New images are recognized by projecting in to the face space. The distance to face space is first calculated and a threshold is applied to check to see if the presented image is a face. This distance, d, is calculated by using the distance between the input image $\vec{I}$ and the reconstructed image $\vec{I}_r$: $d^2 = \left\| \vec{I} - \vec{I}_r \right\|$. If this distance is above the predefined threshold, the distance to the nearest known face is then calculated: $d_e = \left\| \vec{\Omega}_k - \vec{\Omega}_I \right\|$, where $\vec{\Omega}_I$ is the weight of the input image and $\vec{\Omega}_k$ is the weight of its nearest neighbour in the training set. If this is below a threshold the face is classified as the k th individual, otherwise it is classified as an unknown face.

Their ideas were tested on a database of 16 subjects whose faces were captured under varying illumination, pose, and scale (2500 images in all). The face recognition system they built was reasonably robust to lighting changes. However, the system performance was dramatically affected by changes in the size of the face in the test image.

2.2.2 Generalised Object Recognition

Murase and Nayar [69] realized that the work of Sirovich and Kirby [65], and Turk and Pentland [67] lay within the domain of pattern classification and did not address the problem of learning complete parameterized models of objects. A more general recognition system would have to be able to recognize an arbitrary object under different poses and illumination conditions. To address this, Murase and Nayar developed a representation that allowed for discrimination among objects and pose estimation. They also proved that an Eigenspace representation of an image is optimal in the correlation sense, that is the distance between two images in the Eigenspace corresponds to their similarity under the l_2 norm.

In their study, images of objects were obtained by placing an object on a turntable that was illuminated by a light controlled by a robotic arm (see Figure 2-9). By varying the degree of rotation (θ_1 in Figure 2-9) and the angle of lighting (θ_2 in Figure 2-9), a parameterized set of images was acquired. These images were normalized in scale and brightness, and an object Eigenspace was built. Since the Eigenspace was built from images of an object whose viewing parameters were varied, Murase and Nayar called the resulting Eigenspace a parameterized Eigenspace. A parameterized Eigenspace was built for each object of interest.

Consecutive images in the training set are quite similar, as they correspond to small changes in the viewing parameters, thus the projections of consecutive views in the object Eigenspace end up being close together. One can imagine that continuously varying these viewing parameters results in a continuous movement of the projection of the images in the Eigenspace. The result of continuously varying all parameters over their entire ranges is a manifold traced out in the Eigenspace.

When a test image is projected into the parameterized Eigenspace, the viewing parameters can be estimated by finding the closest point on the manifold. Since only a discrete set of views are available, the manifold is approximated by cubic spline interpolation between sample points. This is only valid provided that the object actually belongs to the

Eigenspace. Thus a method for determining which object Eigenspace should be used for parameter estimation is needed.

The problem of selecting the particular object's Eigenspace was addressed by constructing a universal Eigenspace from all images of all the objects. The result is an Eigenspace containing a number of manifolds, one for each object. When a test image is projected to the global Eigenspace, determining the closest manifold provides an indication of which object is in the image. Since the universal Eigenspace represents a number of objects, it is best suited for object discrimination, so once an object is identified, it is projected into its specific object Eigenspace for accurate parameter estimation.

Based on these ideas, Murase and Nayar built a recognition system. This worked well on both objects with uniform reflectance properties and objects with complex appearance characteristics. They determined that a universal Eigenspace with 10 dimensions was sufficient to get near perfect recognition performance in their case. However, their recognition performance was based on two assumptions about the test image: 1, that it contained a well segmented object; 2, that the object was free of occlusion. These two assumptions are the main limitations in real world applications.

2.3 Further research into appearance models

After Murse and Nayer, there are many researchers concentrating on appearance-based methods and tried to overcome the drawbacks of the original method. Most of them moved towards a local appearance space.

2.3.1 Handling occlusion

A major problem with the Eigenspace-based approach is the lack of ability to handle more than one object in the scene, creating the possibility of occlusion. Huang and Camps [70] proposed segmenting the input images into parts and using the appearance of the parts and

their relationships to identify objects in the scene as well as their pose. However, as is well known, segmentation of a scene in this manner is difficult, and in many respects deflects from the elegance of the Eigenspace approach.

Realizing the drawback of their original work on handling occlusion, Nayar et al.[71] adopted the method of recognizing an object in an image based on only a subset of its pixels. Rather than randomly select the subset of pixels or based on some ad-hoc heuristics, they derived several criteria for selecting the subset of image pixels that maximize recognition rate by analyzing the sensitivity of the subspace to image noise. Their research is based on the fact that the image noise degrades the estimates of the subspace projection and the reconstructed image and the degradation also depends on the properties of the rows of A, where A is the orthonormal matrix whose columns are eigenvectors of the training set. They tried to derive the properties that the rows of A should satisfy in order to minimize the degradation due to noise.

After analysing the sensitivity of the rows of A, they derived a heuristic to give priority to under-represented directions: pixels that correspond to diagonal elements of H which have large magnitudes should have higher selection priority, where $H = AA^T$ is the projection matrix. This criterion is then used in two algorithms: window selection and pixel selection. The first algorithm aims mainly to handling occlusion; it automatically selects a square window within an image as the pixel subset. The second algorithm selects the subset from the entire image, i.e., the pixels are not restricted to lie within a local region. They tested the algorithm using images with random noise and the results are better than the random select one. But we can imagine that if the noise or occlusion are in the position of the selected square windows or pixels, it won't work well.

Leonardis and Bischof [15] [16] handled occlusion by randomly selecting image points from the scene and their corresponding points in the basis eigenvector. Their method used a hypothesize-and-test paradigm, where a hypothesis is a set of image point locations (initially randomly generated) and the Eigenspace prototype. The hypothesis-and-test paradigm is conducted as follows: a robust estimator determines the points in the

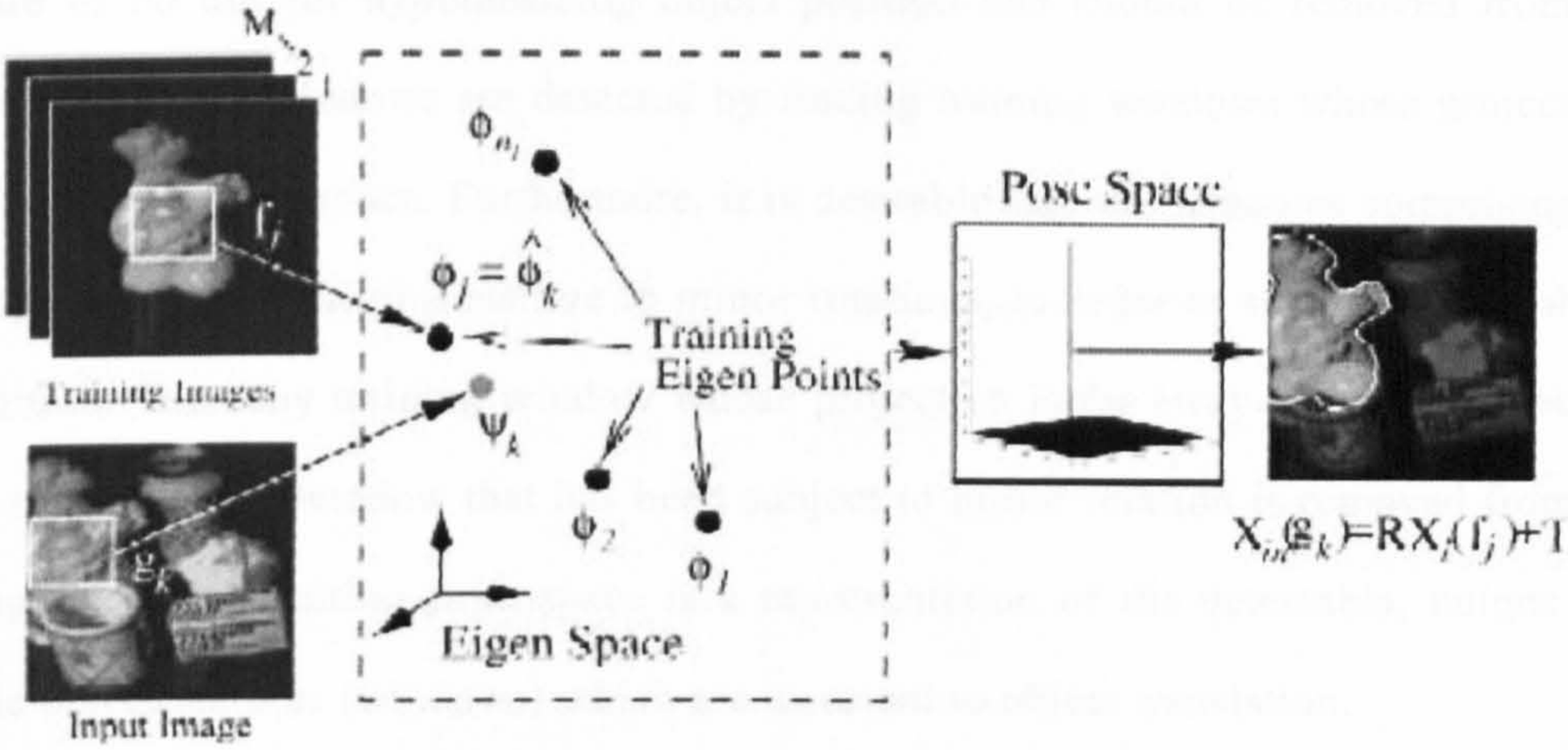


Figure 2-10 Eigen window technique. Source from [73]

Eigenspace which result from the projection of the data vector onto the Eigenspace; the obtained points are then used to create a hypotheses which is analyzed based on error and the number of compatible points; good hypotheses are then chosen based upon the minimum description length principle [72]. This method has the ability to reconstruct unseen portions of the objects in the scene, and is a more viable solution in our context.

2.3.2 Runtime segmentation

A drawback of an appearance-based part representation is that the input images must be segmented at runtime before recognition can occur. This limits the class of objects that can be recognized to those that can be segmented reliably. Most free-form objects do not lend themselves to easy repeatable segmentation.

Ohba and Ikeuchi [73] were able to handle this problem using eigenwindows (see Figure 2-10). The eigenwindows encode information about an object's appearance for only a small section of its view. For each training image, gradient statistics are calculated for windows surrounding each pixel. This gradient information can be used to determine how detectable each window is. An Eigenspace is then built from all the windows in the training images that are above a set detectability threshold. Since many of these windows will be quite similar, for example the windows along the edge of a rectangular surface,

they are of no use for hypothesizing object position and should be removed from the training set. These windows are detected by finding training windows whose projections are close in the Eigenspace. Furthermore, it is desirable that the windows comprising the Eigenspace are relatively insensitive to minor rotations, in order to achieve stable object recognition. Thus any training window whose projection is far away from the projection of the same training window that has been subject to minor rotation is removed from the training set. The resulting Eigenspace is a representation of the detectable, unique and reliable object features (windows) which are invariant to object translation.

Recognition is performed by projecting each test image sub-window into the Eigenspace. The training window whose Eigenspace coordinates are closest to the test images sub-window projection is used to vote for a particular object at a particular position. The final set of votes is then used to determine which objects appear where in the image.

The nice feature of this approach is that the windows that are chosen for building an Eigenspace capable of reliable recognition are tuned to the specific training set. Thus ambiguous and unreliable windows are automatically discarded. These measures are used to omit eigenwindows from the training set if they are hard to detect (detectability), have poor saliency (uniqueness), or are sensitive to noise (reliability). Using the local eigenwindows, they were able to identify multiple objects in cluttered scenes.

2.3.3 Dealing with the ambiguity between objects

The Eigenspace-based approach faces problems once the features available from a single view are simply not sufficient to determine the identity of the observed object. Such a case happens, for example, if there are objects in the database which look very similar from certain views or share a similar internal representation (ambiguous objects or object-data); a difficulty that is compounded when we have large object databases.

Sipe and Casasent [74] described a system in which individual views of an object were modelled as points in Eigenspace and objects represented by linear interpolation between

these points. The resulting data structure was called a feature space trajectory (FST). View planning was accomplished by learning for each pair of objects the most discriminatory viewpoint in an off-line training phase. A viewpoint is highly discriminating if the two FSTs of the inspected object pair are maximally separated.

Borotschnig and Paletta [75] presented a method within an active vision framework for recognizing objects which are ambiguous from certain viewpoints. Depending on the uncertainty in the current object classification the recognition models acquired new sensor measurements in a planned manner until the confidence in a certain hypotheses reached a pre-defined level. Otherwise, another termination criterion was used. They found that the number of dimensions of the feature space can be lowered considerably if active recognition is guiding the object classification phase. Even objects sharing most of their views can be disambiguated by the active movement that placed the camera such that the differences between the objects become apparent.

Murase and Nayar [76] use illumination planning to distinguish objects from each other in appearance. The goal of the study was to determine illumination parameters to maximize the difference in appearance between objects. The resulting source direction can than be used to optimize the performance of the recognition system. The study produced graphs of minimum distance between object curves in Eigenspace as a function of light source direction. In all cases the experimentally determined optimal source was found to have a higher recognition rate in the presence of noise and segmentation errors.

For all the three approaches above, the recognition is done in a structured environment. In this controlled environment, vision systems can be used to perform a variety of tasks, such as inspecting manufactured parts, recognizing objects and sorting them, or aiding a robot in assembly operations. However, in other applications in non-controllable environment, e.g., outdoor scene, the above method is difficult to apply.

2.3.4 Similarity measurements

In this work, a key issue is measurement of similarity between an unknown scene object with several learnt instances of the same and different objects represented by a manifold. Fitzgibbon and Zisserman [77] developed 3 types of new similarity measure that are invariant to affine-transformations,

- $d(x_1, x_2)$, a distance between two images instances x_1 and x_2 which is invariant to deformations;
- $d(x, S)$, a distance between a point x and set of points S that contains exemplars of the deformations;
- $d(S_1, S_2)$, a distance between two sets of images S_1 and S_2.

They started by the defining the first similarity measure, $d(x_1, x_2)$, as the negative log likelihood $p(x_1, x_2)$ that both observations x_1 and x_2 are samples of the same $\tilde{x}$, a "true" datum:

$$d(x_1, x_2) := -\log p_{MAP}(x_1, x_2) \tag{2-3}$$

In the above Equation, $p_{MAP}(x_1, x_2)$ is the MAP (maximum a posterior) estimate of the joint likelihood $p(x_1, x_2)$. Assuming x_1 and x_2 are generated by applying transformations a_1 and a_2 to a true datum $\tilde{x}$, Fitzgibbon and Zisserman proved that $p_{MAP}(x_1, x_2)$ can be calculated by:

$$p_{MAP} = \max_{a_1, a_2, \tilde{x}} p(x_1 \mid \tilde{x}, a_1) p(x_2 \mid \tilde{x}, a_2) p(a_1) p(a_2) p(\tilde{x}) \tag{2-4}$$

Then the distance can be rewritten as a sum of negative log likelihoods

$$d(x_1, x_2) = \min_{a_1, a_2, \tilde{x}} E(x_1 - T(\tilde{x}; a_1)) + E(x_2 - T(\tilde{x}; a_2)) + E(a_1) + E(a_2) + E(\tilde{x}) \tag{2-5}$$

where $T(\tilde{x}; a)$ represents the affine transformation of the image. Fitzgibbon and Zisserman referred to this distance as the manifold distance between two points

They showed that this *manifold distance* has advantages over several alternative definitions in the literature. They are "one-sided" distance [78], "two-sided" distance [79], and "symmetric transfer distance"[80], shown in Figure 2-11. The first one is not

symmetric at all; the second one can sometimes make distances between disparated objects arbitrarily small; the third one does not have the first two problems but may still not include the prior terms.

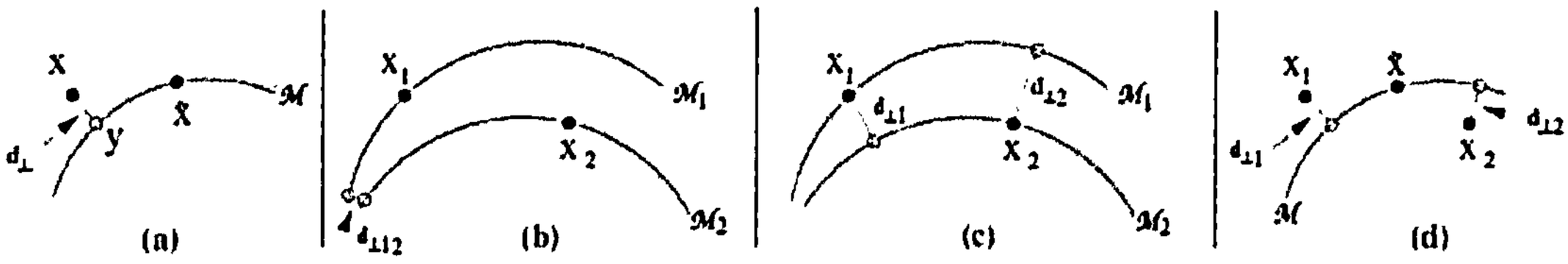


Figure 2-11 Several definitions of distances between sample points x_1 and x_2. (a) One-sided distance (Transfer distance); (b) Two-sided distance; (c) Symmetric transfer distance; (d) manifold distance. Source from [77]

Followed by a definition of the point to point distance, they then defined the point to subspace distance and the distance between subspaces, which they called the *joint manifold distance*. A linear subspace of images was defined by a mean image m and a set of basis vectors M. Any image in the space can be linearly parameterized by a vector u yielding the image set $S = \{m + Mu \mid u \in U\}$. The point-to-subspace distance is defined by

$$d(x_1, S) = \min_{u,a} \left\| T(m + Mu; a) - x_1 \right\|^2 + E(a) + E(u) \tag{2-6}$$

The joint manifold distance between two subspaces S and T, where $S = \{m + Mu \mid u \in U\}$ and $T = \{n + Nv \mid v \in V\}$ is:

$$d(S,T) = \min_{u,v,a,b} \left\| T(m + Mu; a) - T(n + Nv; b) \right\|^2 + E(a) + E(b) + E(u) + E(v) \tag{2-7}$$

All the three types of distances can be computed by calculating the Taylor expansion of $T(\cdot;\cdot)$, yielding the subspace analogue of tangent distance. In their work, an application of the joint manifold distance was to compare image sequences in a video and to find out whether the two sequences contain the same character.

2.4 Recognition in infrared imagery

The methods discussed in the previous sections are general in that they can be applied to both visual and infrared imagery. The performance in visual and infrared imagery using these general methods has been compared (see section 2.4.1) and methods designed specially for infrared scenes are reviewed (see section 2.4.2 and 2.4.3) in this section. A review of fundamental of infrared imaging including infrared radiation, atmosphere absorption, infrared image calibration is in Appendix A.

2.4.1 Performance comparison between visual and infrared imagery

Thermal images are obtained by sensing the radiation in the infrared spectrum, which is either emitted or reflected by the object in the scene. The (primarily) emitted radiation of LWIR energy has a strong dependence on internal composition, properties, and the state of the object. This dependence brings advantages and problem when analyzing thermal images.

The processing of infrared images presents some advantages with respect to images in the visible domain [3] .

(i) Infrared (particularly LWIR) sensors are to a lesser extent dependent on different weather and illumination conditions than visible wave sensors: even day or night snapshots of the same scene are every similar, thus reducing the range of situations to be taken into account [81].

(ii) The use of infrared images can be a solution to the problem of detecting objects that feature different colours or textures and of avoiding camouflaging patterns such as shadows.

Unfortunately, there are problems encountered in thermal computer vision that are not encountered in the visible spectrum [11].

(i) Internal characteristics of an object that are not evident in a normal picture may be significant in a thermal image. For instance, a thermal image, captured when the engine of the car has been activated, is different from one collected when the engine is still.

(ii) Aspects of the environment can affect the thermal image. For example, the ambient air temperature in an outdoor environment can affect skin temperature and hence the thermal image of a car. Similarly, wind and sun can expose different sides of a car to different thermal loads and hence create local variations in appearance across the skin surface.

(iii) In contrast to visual images, the images obtained from an infrared sensor have an low signal to noise ratio (SNR), which results in a degradation of information for performing detection or tracking tasks.

Several authors have compared the performance of infrared and visual imagery for recognition. An equivalent recognition performance has been shown in [12] using a low quality pyroelectric infrared sensor.

In later research, using a LWIR microbolometer which is sensitive through the range 8-12um, D.A. Socolinsky, et al. [13] studied multiple appearance-based face recognition methodologies, including PCA, LDA(linear discriminate analysis), LFA(local feature analysis) and LCA, on visible and thermal infrared imagery. Their analysis reveals that under many circumstances, using thermal infrared imagery yields higher performance, while in other cases performance in both modalities is equivalent. For example, their results showed a reduction in the residual error by over 30% for identification applications and lowering of the EER (Equal error rate) by over 20% using thermal imagery. Furthermore, they demonstrated that combined use of both imaging modalities results in even higher performance, with identification errors dropping by more than 45% and EERs lowering more than 40%.

This better performance can be partly explained by the following statements.

(i) Thermal infrared imagery of objects is nearly invariant to changes in ambient illumination.

(ii) The perceived intensity of radiation from a blackbody is independent of surface orientation.

Although this performance comparison research is conducted in the scenario of human face recognition, in which the temperature and illumination are restricted to a relevant small scale when compared to an outdoor or remote sensing environment, the methods they used for recognition are general. It implies that even if we don't use special features of an infrared image (which will be explained in a later section) and used the same method for infrared and visual image recognition, infrared image recognition may at least perform equivalently to visual image recognition.

2.4.2 Thermophysical invariants

As the wavelength of the sensor transducer passband increases, e.g. in infrared imagery, emissive effects begin to be the dominant mode of electromagnetic energy exitance from object surfaces. The emitted radiosity has a strong dependence on internal composition, properties and state of the object. This dependence may be exploited by specifying image-derived invariants that vary only if the physical properties vary. These invariants are called thermophysical invariants.

Thermophysical invariants can be derived from a function which is based on the principle of conservation of energy at the surface of the imaged object (see Figure 2-12)

$$W_{abs} = W_{lost} = W_{cnd} + W_{st} + W_{cv} + W_{rad} \tag{2-8}$$

where W_{abs} is the energy absorbed, W_{lost} is the energy lost, W_{cnd} denotes the energy conducted from the surface into the interior of the object, W_{st} is the stored energy, W_{cv} is the energy converted from the surface to the air, and W_{rad} is the energy lost by the surface to the environment via radiation. Among the energy components above, Nandhakumar and Aggarwal [82] believe that the conduction heat transfer W_{cnd} is highly dependent on the

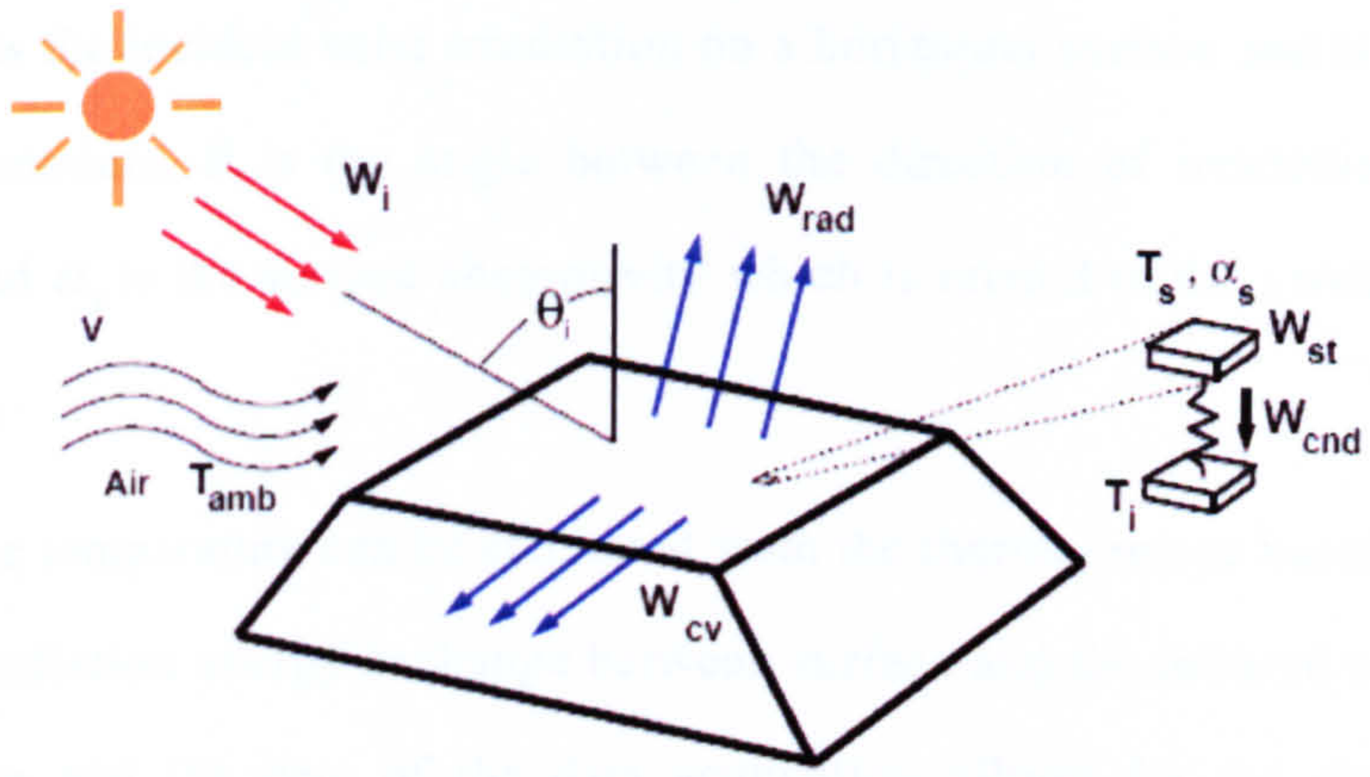


Figure 2-12 Energy exchange at the surface of the imaged object. Incident energy is primarily in the visible spectrum. Surface loses energy by convection to air, and via radiation to the atmosphere. An elemental volume at the surface is shown. Some of the absorbed energy raises the energy stored in the elemental volume, while another portion is conducted into the interior of the object. Source from [84]

properties of the object surface and expect it to be a useful feature for distinguishing objects from each other. To minimize the feature's dependence on differences in absorbed heat flux, a normalized feature was defined to be the ratio $R = W_{cnd} / W_{abs}$.

To compute W_{cnd} and W_{abs}, they formulated a thermophysical model to allow integrated analysis of thermal and visual imagery of outdoor scenes. Surface temperature of the viewed object is inferred using the thermal image. This information along with knowledge of ambient air temperature allows for the estimation of the radiation heat loss at the surface using the equation:

$$W_{rad} = \varepsilon\sigma\left(T_s^4 - T_{amb}^4\right) \tag{2-9}$$

where σ donate the Stefan Boltzman constant, T_s is the surface temperature of the images object, and T_{amb} is the ambient temperature. Convection heat loss is computed with the knowledge of wind speed, air temperature, and surface temperature using the equation:

$$W_{cv} = h\left(T_s - T_{amb}\right) \tag{2-10}$$

where h is the average convection heat transfer coefficient, and depends on the properties of the surrounding air and on the geometry and the nature of the objects surface. The energy absorbed by the surface is given by:

$$W_{abs} = W_I \cos\theta_I \alpha_s \tag{2-11}$$

where W_I is the incident solar irradiation on a horizontal surface and is given by available empirical models, θ_I is the angle between the direction of irradiation and the surface normal, and α_s is the surface absorptivity which is related to the visual reflectance ρ_s by $\alpha_s = 1 - \rho_s$.

The surface temperature can be estimated from the thermal image based on an appropriate model of radiation energy exchange between surface and the infrared camera. Knowledge of the time and the date of the data acquisition allows for the determination of the magnitude and direction of solar irradiation. The visual image provides surface reflectivity and relative orientation, which when combined with solar irradiation information, allows for the estimation of the absorbed heat flux. Finally, using the surface heat balance model, the conductive heat flux and hence the ratio R is computed (here they ignore the contribution of W_{st} because it is too small compare with W_{rad}).

They noted that the ratio R provides significant information about surfaces studied and is useful for discrimination between the types of objects in the scene. For example, the values are lowest for vehicles, highest for vegetation and in between for buildings and pavements.

The above technique requires a priori knowledge of several surface and scene parameters such as emissivity, wind speed, etc., which in many applications are unavailable. Even in those situations where such information is available, the thermophysical feature, R, was found to be only weakly invariant. The range of values of R for each class was observed to vary with time of day and season of year. In addition, the feature R was able only to separate very broad categories of objects but lacked the specificity to differentiate between different models of vehicles.

Nandhakumar et al. [83] proposed an improved formulation for establishing thermophysical features, wherein the feature was constrained to be invariant to affine

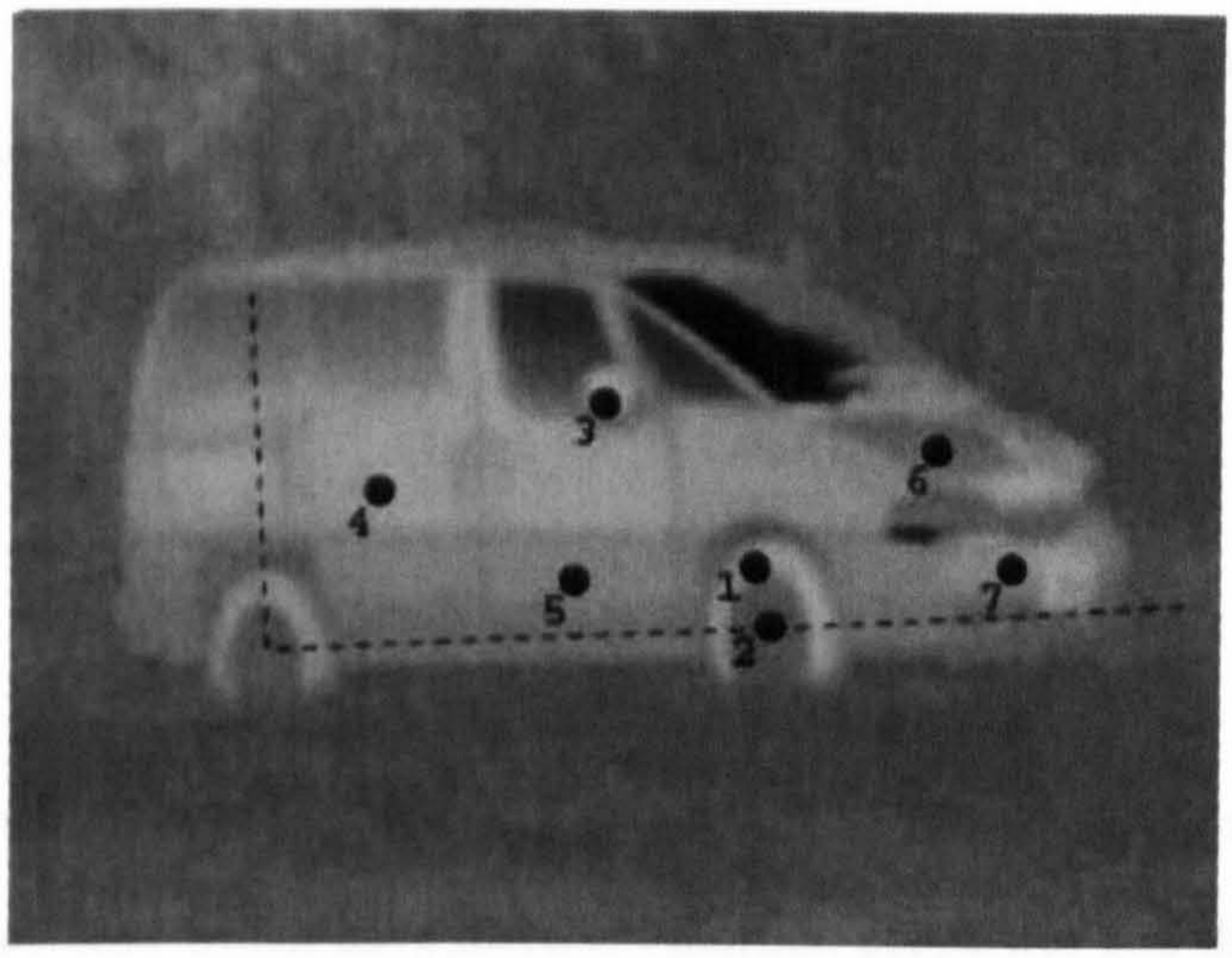


Figure 2-13 The van object type with points selected on the surface with different material properties and/or surface normal. Point 1: Vulcanized Rubber, 2: Aluminium Alloy, 3: Polystyrene-like polymer, 4: Steel, 5: Polypropylene-like polymer, 6: Steel, 7: Polypropylene-like polymer. Source from [84]

transformations of the driving (scene) conditions. In their work, instead of computing the conducted energy W_{cnd} from the energy equation, they compute it using

$$W_{cnd} = -k\, dT/dx \tag{2-12}$$

where k is the thermal conductivity of the material, and x is the distance below the surface. Also they compute the energy stored in the surface by

$$W_{st} = C_T\, dT_s/dt \tag{2-13}$$

where C_T denotes the lumped thermal capacitance of the object and is given by $C_T = DVc$, D is the density of the object, V is the volume, and c is the specific heat.

Given Equation (2-9) to Equation (2-13), Equation (2-8) can be rewritten in the following linear form:

$$a^1 x_1 + a^2 x_2 + a^3 x_3 + a^4 x_4 + a^5 x_5 = 0 \tag{2-14}$$

where

$$\begin{aligned} a^1 &= \cos\theta_I & x_1 &= W_I \alpha_s \\ a^2 &= -\sigma\left(T_s^4 - T_{amb}^4\right) & x_2 &= \varepsilon \\ a^3 &= -\left(T_s - T_{amb}\right) & x_3 &= h \\ a^4 &= C_T & x_4 &= -dT_s/dt \\ a^5 &= kT_s & x_5 &= 1 - T_{\text{int}}/T_s \end{aligned} \tag{2-15}$$

Thus for each pixel in the thermal image Equation (2-14) defines a hyperplane in 5D space expressed by the vector $\vec{a} = [a^1\ a^2\ a^3\ a^4\ a^5]$. They also proved that the ratio of the determinant of two point set $I = d(a_i a_j a_l a_m a_n) / d(a_p a_q a_r a_s a_t)$ is the thermal physical invariant in different scene conditions, where $d(a_i a_j a_l a_m a_n) = \begin{vmatrix} \vec{a}_i \\ \vec{a}_j \\ \vec{a}_l \\ \vec{a}_m \\ \vec{a}_n \end{vmatrix}$.

The choice of the two set of points must satisfy several criteria, e.g., the points in each set must be from the material which have different thermal properties and the two points set must have at least one point different from each other. Even the criteria is satisfied, the set of points then has to be selected empirically based on the smallest variance under different environmental conditions.

Later, J. Michel, et al. [84], employed a hypothesize-and-verify strategy. This strategy contained 3 stages:

(i) Hypothesise the object class using geometric invariants

(ii) According to the hypotheses, assign thermophysical properties to the object of interest and compute the thermophysical invariants

(iii) Compare the computed thermophysical feature with the one from a model prototype to verify the hypothesis.

The thermophysical invariants they derive do not have any physical meaning, but are algebraic invariants formed by algebraic elimination of five invariant functions. In stage 2, they chose four points of the image to calculate the invariants. Figure 2-13 shows the points selected on the surface of different materials and orientation from which the four points are chosen.

Although the derivation of the features is constrained so that the values should be invariant from one scene to another, class separation is still not explicitly incorporated.

Hence, practically, the use of this approach for recognition requires searching all possible features for the best separation. It is not clear that a solution will always exist.

2.4.3 Other recognition techniques in infrared imagery

There are many other techniques dealing with infrared recognition. This section will discuss some of them, e.g. statistical and motion based techniques.

A statistical model for segmentation

For segmentation of human faces, Eveland, et al. [85] classified pixels in indoor scenes as belonging to one of the three classes: exposed skin, covered skin and background. They used a probabilistic model to segment these three regions and take a Bayesian approach to derive this model:

$$P^t(c_j \mid r) = \frac{\pi_j^t P_j^t(r)}{\sum_{i=1}^{n} \pi_i^t P_i^t(r)} \tag{2-16}$$

where $P_j^t(r)$ is the class-conditional density of radiance of radiance for class c_j, and π_i are the class priors at time t. This allows us to compute for any given pixel in the current frame the likelihood of belonging to each class.

It should be noticed that the class-conditional densities used in the training part are dependent on a good calibration, and normal indoor conditions. The outdoor skin distributions vary markedly from individual to individual. This was due to

(i) differences in thermal exposure of the skin to the sun as the weather changed over the day, and

(ii) difficulty in keeping a good calibration as the sun affected the camera's cooling electronics.

To apply their technique outdoors, we should have to improve the calibration process and perhaps help initialise the skin densities from another source.

Recognition through simulation

Many FLIR simulation efforts have been undertaken with the goal of training and testing ATR algorithms and predicting performance. In the work of A.D.Lanterman, et al. [86], using the pattern theoretic Grenander/Bayesian approach to the ATR problem, simulation provides the heart of the ATR algorithm itself.

Their approach represents a dramatic departure from a traditional machine vision algorithm which maintains a conceptual separation between "low-level" vision (edge detection, segmentation, etc.) and higher levels of inference (classification). Instead of performing separate steps of segmentation, feature extraction, etc., they estimated the configuration of targets directly from the measured data.

They take a Bayesian approach in which a hypothesized scene, simulated from the emissive characteristics of the hypothesized scene elements, is compared with the collected data by a likelihood function based on sensor statistics. They built deduction algorithms around jump-diffusion processes (a searching process) that provides the dynamic flexibility to accommodate higher and lower complexity scenes. Jump-diffusions are inherently discrete and continuous in the nature of their search, and therefore accommodate the very different continuous and discrete nature of image understanding. The jump deals with changes of target type and number and the diffusions accommodate the continuous parameters such as the positions and orientations of targets.

Pattern theoretic algorithms based on jump-diffusion processes accommodate geometric variability (target appearances vary with their orientations and positions) and complexity/scene variability (number of targets not known in advance). Their later work [87][88] extended to better accommodate the thermodynamic variability by summarizing the thermodynamic state of targets with a parsimonious set of variables that become nuisance parameters in the Grenander/Bayesian formulation. One of the contributions of this algorithm is that it required no prior knowledge of the intensities of the targets or the

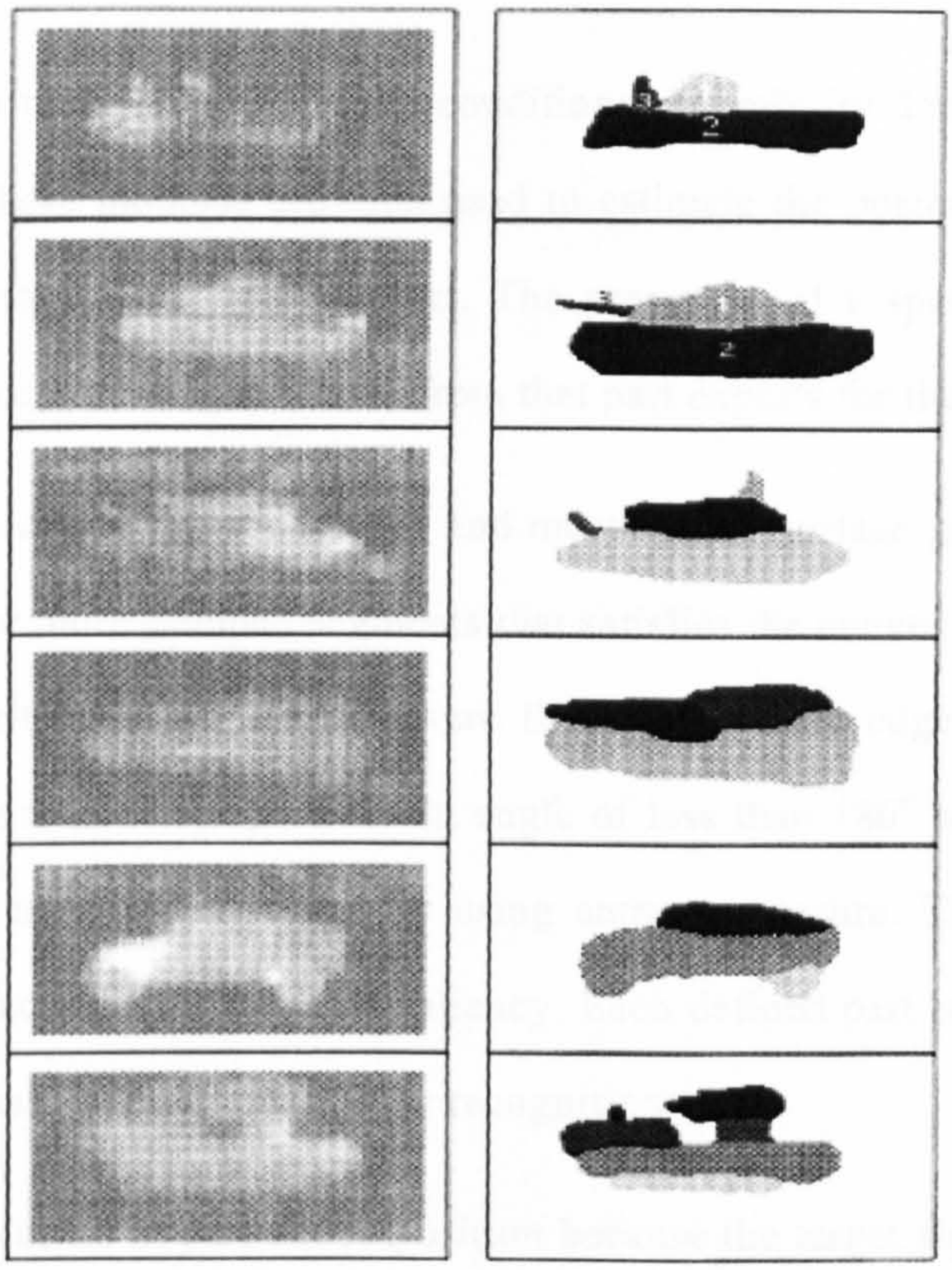


Figure 2-14 Typical FLIR images of targets used in recognition experiments in [90]. The figure also shows that parts identified for various targets. Sourced from [90]

background. A simple likelihood model was used to incorporate radiant nuisance parameters.

Recognition by Image Features

Nair and Aggarwal [89] [90] proposed a hierarchical recognition strategy that uses salient object parts as cues for classification and recognition for FLIR images. The lower level classifiers recognize the class of the input object, while the higher level classifiers recognize the specific type of object. At each level, targets are recognized using their parts and thus each target classifier is made up of models, each of which is an expert on a specific part of the target. Each modular expert (or classifier) is trained to recognize one part under different viewing angles and transformations. When presented with an input target part, each expert provides a measure of confidence of that part belonging to the target that the expert represents. In their work, the modular experts is built using Bayesian

approach, where each model represents the conditional probability density function of a part, and outputs of these modules are then used to estimate the posterior probability of the input part belonging to a specific object. The presences of a specific target in the image is decided by accumulating evidence from that part experts for that targets.

A part of an object is defined as the largest and most salient surface generated from the combinations of one or more grouped segments that satisfies the convexity constraint (see Figure 2-14). Convexity constraint makes sure that groups only edge segments whose tangents at the common corner point from an angle of less than 180° when viewed from inside the target. Saliency is determined by using entropy measure. The lower the cost associated with a surface, the higher is its saliency. Each defined part is than represented using Zernike moments [91] up to order 8 for recognition.

This method has difficulties in part decomposition because the target signatures vary and distribution of parts is not readily obtainable. In addition, high-order Zernike moments are sensitive to digitization errors, minor shape deformations and background noise [92].

Sun and Park [4] extract features from the boundary of the object to recognize non-occluded and partially occluded object in FLIR images. The boundary of the object, which is obtained by using the Canny edge detection that include the Gaussian smoothing process to reduce the noise effect, is been partitioned into upper, lower, left and right regions. Four global features of the entire boundary and four local features of the each part are defined from the redial function of the boundary. Those features, includes amplitude variation, skewness etc., are invariant to scaling and rotational transforms. For classification, they trained multilayer perceptron (MLP) [93] [94] [95] for each feature set. When a new image comes in, the recognition result is obtained by averaging the results of the four MLPs.

2.5 Conclusion

In this chapter, we have reviewed general object recognition techniques and those in infrared imagery. Generally, an object recognition system consists of two stages: one is constructing a model library from certain descriptions of the objects; the other is finding a correspondence between certain features in the image and similar features of the object. In the text, we refer the first procedure as ***Modelling*** and the main part of second procedure as ***Matching***.

Looking for invariants is a straight forward approach in object recognition. If certain descriptions of an object are identical independent of viewing directions, lighting conditions, etc., then the features extracted from the input image can be compared directly with those features stored in the object model. The research of invariants goes from invariants under affine transformation to perspective transformation, from planner objects to 3D objects, from polygons to free-form objects, and from geometric invariants to the combination of geometric and illumination invariants. As the wavelength range of the sensors extended, more invariants can be extracted. In infrared imagery, thermal physical invariants are defined for object recognition based on the principle of conservation of energy at the surface of objects. A common drawback of the invariants approach is that most of those invariants are derived from some local parts of objects, e.g., corners, edges, local shading, etc.. , This approach could work well for applications under a controllable environment, e.g., recognition of industrial parts. However, in applications under bad imaging conditions, where it is difficult to correctly identify those critical local parts and features, this approach will face a challenge.

The chosen of techniques for ***Matching*** largely depends on what strategy used in ***Modelling*** stage. If the descriptions of the objects derived from modelling stage can be represented by a set of discrete features, an interpretation tree can be used to find the correspondence between the model and the features in the scene. Geometric constraints between image features can be used to test feasibility and prune large portion of the tree. If in the modelling stage, the relations among features are more strongly emphasized,

those relation can be part of core components in the model. For example, the object can be represented as a graph with each node representing a feature and each graph edge represent relations among the features. In this case, the matching in the recognition stage becomes a graph matching problem. In other modelling methods where the features in the images and the models can be normalized so that they can by represented in the same metric space, we can use classification method to do the matching. In these approaches, the features for the object can be represented as a point in multi-dimensional space. Examples of the classification methods are Nearest Neighbour Classifiers, Bayesian Classifiers, Neural Nets, etc..

The appearance based object recognition receives a great attention in the literature because it has several advantages, e.g., since it is kind of attracting 'global' features of the object, any local damage won't affect the results as it does to many other recognition method; the method does not restrict to any particular type and shape of objects, etc.. Many researches following this approach have made effort to improve the performance of the recognition system when the input image is with noise and occlusion by concentrate on local appearance space. These include methods intending to focus on the fixed featured regions and on verified random selected regions. In this thesis, we adopt a random sampling method to deal with noise and occlusion. The implementation and experimental result of this method can be found in Chapter 3. The appearance based method has been used in object recognition in infrared imagery. However, no modifications have been made to account for the different characteristics of infrared imagery from visible imagery. In Chapter 5, we propose an appearance based object recognition system especially for infrared imagery by modelling the changes in thermal state.

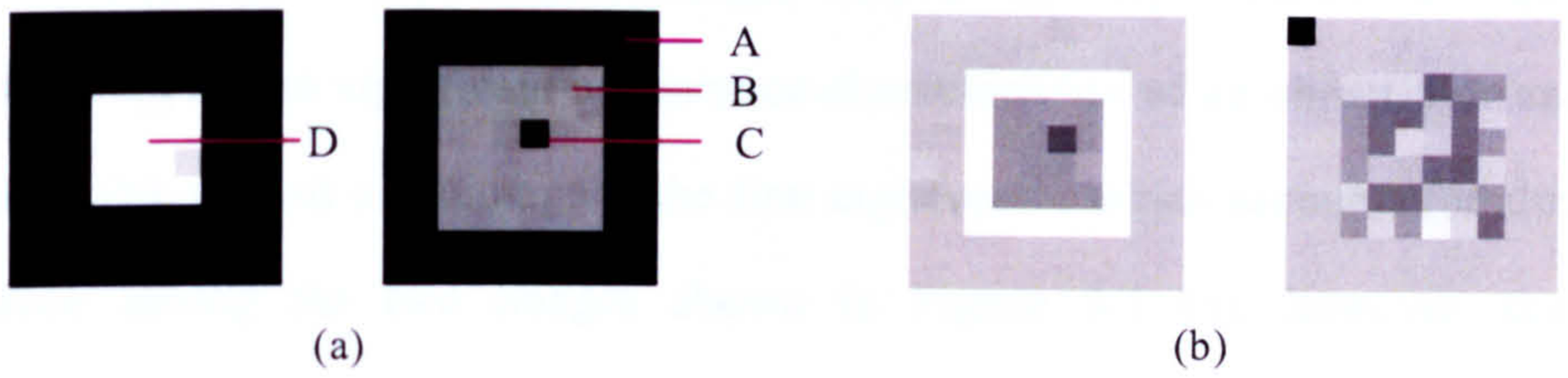
Chapter 3

Eigenspace Based Recognition and its Improvements

The idea of using an Eigenspace in object recognition comes from a question--*what aspects of the object stimulus are important for identification?* The information theory of coding and decoding images may answer this question in that it gives insight into the information content of images, emphasizing the significant local and global "features". Such features may or may not be directly related to our intuitive notion of object features such as the wheels of cars, eyes of humans or wings of birds.

One approach to extracting the information contained in an image of an object is to somehow capture the variation in a collection of images, independent of any judgment of features, and use this information to encode and compare individual images. This approach is based on principal component analysis and to find the principal components of the distribution of objects, or the eigenvectors of the covariance matrix of the set of images, an image is treated as a point (vector) in a very high dimensional space.

The eigenvectors can be thought of as a set of features that together characterize the variation between images. Each image location contributes more or less to each eigenvector, so that we can display the eigenvector as an image. We call this image an eigenimage. The eigenvectors are ordered, each one accounting for a different amount of the variation among the images. Each eigenimage deviates from uniform gray where some local part differs among the set of training images; it is a sort of map of the variations



(a) (b)

Figure 3-1 An example of Eigenvectors. (a)original images (b)first two Eigenimages

between images. As a very simple example, we consider a set with two images of the same size (see Figure 3-1 (a)), and calculate the eigenvectors for this image set. In the example image set, we define four areas, in which area A is the black edge area in right image where pixel values are the same in both images, areas B and C are the gray area and black spot in right image, and area D is the bright area in left image. Figure 3-1 (b)[1] displays the images of the first two eigenvectors.

The left image in Figure 3-1 (b) is the first eigenvector that shows the main difference between the two images in the image set shown in Figure 3-1 (a). In area A, because there are no difference between two images, the pixel brightness is uniform gray; in areas (B – D) and (D – C), because pixel values are different in original image set, the pixel values in the first eigenimage are from uniform gray to brighter and darker. Also, we see that the eigenimage does not only record the difference between images, but also records how much different they are. For example, in both areas C and (D – C), the pixel values in right image are darker than the ones in left image but the differences in area C are more significant. As a result, the pixel values in area C are further away from the gray value than in area (D – C).

The eigenvectors form a basis for representing individual images in the image set such that each individual image can be represented exactly in terms of a linear combination of the eigenimages. Though a large number of eigenvectors may be required for very

[1] Note that here the pixel values in Eigenimages are not between 0 to 255, as in the original images. They are between -1 and 1. When we display them, we set the minimum value as black, maximum value as white and the values in between as intermediate shades of gray. So the uniform gray in the image means that the pixel value is zero.

accurate reconstruction of an object image, only a few eigenvectors[2] are generally sufficient to capture the significant appearance characteristics of an object. For example, in Figure 3-1 (b), the left eigenimage is the first eigenvector which accounts for almost all the variance among the two images shown in Figure 3-1 (a), however, the right eigenimage represents little. These eigenvectors constitute the dimensions of the Eigenspace of the image set.

If the individual images can be reconstructed by weighted sums of a small collection of characteristic features or eigenimages, this can be an efficient way to learn and recognize objects. We can recognize particular objects by comparing the feature weights associated with known individual images. Each individual image, therefore, would be characterized by the small set of feature or eigenimage weights needed to describe and reconstruct them.

In this chapter, we describe the theory of Eigenspace-based object recognition algorithm based on Murase and Nayar [69] and Turk and Pentland [67]'s work, together with tutorial examples and discussion of implementation (Section 3.1). Then, to strengthen the model when the training set is not big enough and also to recognize object with pose parameters between that of the object in training images, we adopt the idea proposed by Murase and Nayar [68] to interpolation the discrete points in Eigenspace to build a hypersurface as the representation of the object in Eigenspace (Section 3.2). A k-d tree searching algorithm has been implemented to make the recognition more efficiency (section 3.3). To make the result of multidimensional k-d tree search equivalent to find the smallest Euclidean distance, an adjustment of k-d tree algorithm is proposed and implemented. Another problem of appearance-based recognition method is that the recognition results are getting worse if the unknown image is with noise and occlusion. We implement the robust sampling method based on Leonardis and Bischof's work [16] (Section 3.4) which aims to get over this problem.

In object recognition, sometimes it's not enough to give only the object identification but also give the confidence of the recognition result. In this chapter, we propose a

[2] We will discuss how many eigenvectors are sufficient in section 3.1.3.

Probabilistic Eigenspace approach which gives a framework for measuring the confidence as well as identifying the object in unknown images (Section 3.5). Using this framework, test images with a small in-plane transformation of the training images can be identified correctly. We then test those algorithms on both visible and infrared image sets (Section 3.6).

3.1 Basic Eigenspace Based Recognition

A general object recognition scheme consists of two procedures: Learning /Training and Identification/Recognition. The Eigenspace-based object recognition system follows these general steps. It begins with a model building stage where a database of objects is examined and their models developed. Then, given a new image, we compare the image and the models to identify which object is presented in the image.

In the training procedure, first an image set of the object is obtained by varying a wide range of imaging conditions in small increments. Then the image set is normalized in both scale and energy to achieve invariance to sensor magnification and illumination intensity. The Eigenspace for the image set is constructed when all training samples are projected into the Eigenspace to get the individual points corresponding to the training samples.

To recognize the object in an input image, we assume firstly that the object is not occluded and can be segmented from the remaining scene. (We discuss approaches to deal with occlusion and noise later.) The recognition module starts by normalizing the segmented image region in the same way as normalizing the images in the training module. The normalized image is then projected into the Eigenspace obtained from the training module to get the Eigenspace point. After comparing this Eigenspace point with the points from the training module, we finally recognize the object in the input image. The following subsections describe the details of computing the appearance model and discuss some key issues.

3.1.1 The Image Workspace

The appearance model is parameterized by the image acquisition variables including object pose, illumination parameters, thermal signatures, etc. We define these variables as the degrees of freedom (DOF) of the image workspace:

$$q = [q_1, q_2, \ldots, q_k]^T \tag{3-1}$$

where k is the total number of DOF. In a given application, q has lower and upper bounds and its continuous set of values within these bounds map to a continuous domain of $i(q)$. This range of appearance is the image workspace. For example, if in an application, there are 100 poses and 20 thermal variations, then q_1 represents pose variation and had 100 values and q_2 represents thermal variations and has 20 values. The image workspace has 2000 images $i([q_1, q_2])$.

To build Eigenspaces, we regard images as vectors. Let an image be a two-dimensional J by K array of intensity values. Each 2D image can be thought of as a $J*K$, 1D column vector of intensity values by scanning the image conventionally from top to bottom and left to right. In this way, a J by K 2-D image is represented by a $J*K$-dimensional column vector. If we were to consider images of size 128×128, then the dimensionality of the vector space containing the images would be 2^{14}. This representation allows us to do the inner product of two vectors when we project one vector to another.

The raw images in the image workspace are normalized in two ways, scale and energy. In scale normalization, each digitized image is firstly segmented into an object region and a background region. The background is assigned zero energy value and the object region is re-sampled such that the larger of its two dimensions fits a pre-selected image size. Then we force the re-sampled object region to be of the same size. The purposes of scale normalization are firstly, to achieve scale invariance, and secondly, minimize the effect of the background region in the recognition performance[3].

---

[3] If a certain position is in the background region with the same value for all images, the pixel values of that position in the Eigenimages are zero. In the recognition process, whatever the pixel values are in that position in the input image, they do not contribute.

In recognition, assuming that the imaging sensor used for learning and recognition has a linear response, i.e. image brightness is proportional to scene radiance, it is desirable that our recognition system be unaffected by variations in the intensity of illumination or the aperture of the imaging system. This can be achieved by normalizing each image, such that the total energy contained in the image is unity, i.e. $\|I\| = 1$. This can be done by dividing each pixel value by the root of the sum of squares of all pixel values, so that the resultant images become:

$$\vec{i} = [X_1, X_2, \cdots, X_m]^T \tag{3-2}$$

where $X_x = (1/A)\hat{X}_x$. We call A the *Energy normalization factor*: $A = \sqrt{\sum_{x=1}^{m} \hat{X}_x^{\,2}}$, where m is the number of pixels in the image and $\hat{X}_x$ is the pixel value before energy normalization.

3.1.2 Computing the Eigenspace

To compute the Eigenspace, the average $\bar{\mathbf{i}}$

$$\bar{\vec{i}} = \frac{1}{n}\sum_{x=1}^{n} \vec{i}_x \tag{3-3}$$

of all images in the set is subtracted from each image to get the difference image set

$$D = \left[\vec{i}_1 - \bar{\vec{i}}, \vec{i}_2 - \bar{\vec{i}}, \vec{i}_n - \bar{\vec{i}}\right] \tag{3-4}$$

The difference image matrix D is $m \times n$, where m is the number of pixels in each image, and n is the total number of images in the image set.

Next, we define the covariance matrix:

$$C = DD^T \tag{3-5}$$

This matrix is $m \times m$, clearly a very large matrix since a large number of pixels constitute an image. If each image contains 128×128 pixels, the matrix will contain, 214×2^{14}, more than 268 million elements.

The eigenvectors e_i and the corresponding eigenvalues λ_i of C are determined by solving the well-known eigenstructure decomposition problem [96]:

$$\lambda_i e_i = C e_i \tag{3-6}$$

We need a computationally feasible method to find these eigenvectors. If the number of data points in the image space is less than the dimension of the space ($n < m^2$), there will be only $n-1$, rather than m^2, meaningful eigenvectors. The remaining eigenvectors will have associated eigenvalues of zero.

Consider the eigenvectors e_i of $D^T D$ such that

$$D^T D e_i = \lambda_i e_i \tag{3-7}$$

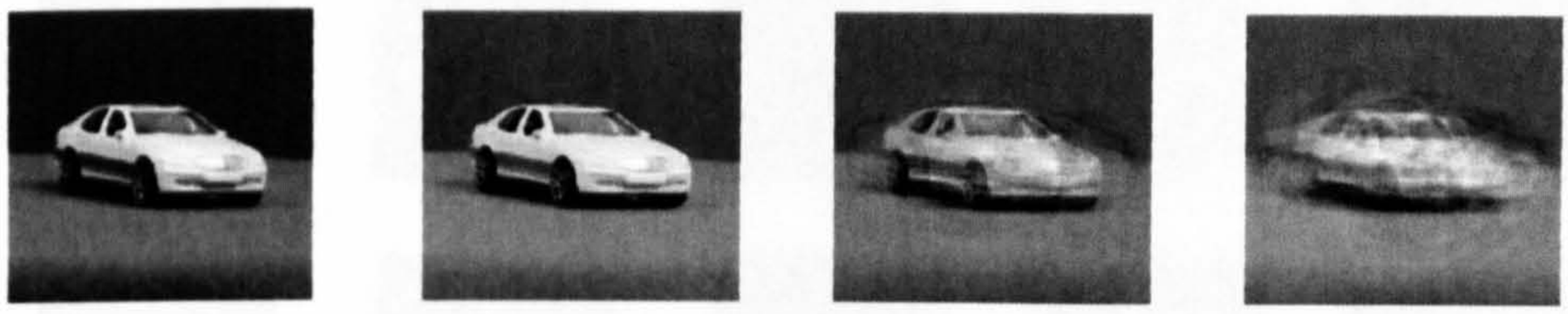
Pre-multiplying both sides by D, we have

$$D D^T D e_i = \lambda_i D e_i \tag{3-8}$$

from which we see that $D e_i$ are the eigenvectors of $C = D D^T$.

Thus we can transform the problem of calculation of the eigenvectors of $D D^T$ to $D^T D$ which is a $n \times n$ matrix. If there are 40 images in the training set and each has 128 by 128 pixels, the calculation will be simplified from calculating the eigenvectors of a 16384×16384 element matrix to a 40×40 element matrix. The number of eigenvectors will be 39.

The necessary number of eigenvectors varies according to the variation of the scale of training set, the content of the training samples, the purpose of the recognition, etc. In face recognition, Turk and Pentland [67] used only 7 eigenvectors based on a training set with 16 face images, which is less than half of all the eigenvectors. The 16 face images they used were comparatively similar to each other. In the training set with 41 views of a car, if we use half of the eigenvectors to reconstruct the image, the result is not acceptable. Figure 3-2(a) shows the original image. Figure 3-2(b) demonstrates that using 40 eigenvectors leads to the best reconstructed image that is identical to the original one. However, as the numbers of eigenvectors involved are decreased, the image



(a) (b) (c) (d)

Figure 3-2 Original (a) and reconstructed images (b)(c)(d) of a car

reconstruction becomes worse and worse. For example, Figure 3-2(c) and Figure 3-2(d) respectively illustrate the reconstructed image using 35 and 20 eigenvectors.

This Eigenspace constructed by eigenvectors as the coordinates is the basis of the whole recognition task. We store them and pass them to the recognition module.

3.1.3 Computing the Eigenspace Points of the Training Samples

The eigenvectors form an orthonormal basis of the vector space spanned by the training vectors, so each training image can be reconstructed in terms of this basis as

$$\vec{i} = \sum_{x=1}^{n} a_x e_x + \bar{\vec{i}} \tag{3-9}$$

where $a_i = (\vec{i}, e_i)$, the inner product of the training image and an eigenvector. The nice feature of this basis is that eigenvectors with larger associated eigenvalues are more significant for accurate reconstruction than eigenvectors with smaller associated eigenvalues. Thus each training image can be approximately reconstructed in terms of the basis as

$$\vec{i} \approx \vec{i}^{\,k} = \sum_{x=1}^{k} a_x e_x + \bar{\vec{i}} \tag{3-10}$$

for some $k \le n$. This subset of eigenvectors spans a vector space referred to as the Eigenspace. The error of the reconstruction can be expressed as

$$E = \frac{\left\| \vec{i} - \vec{i}^{\,k} \right\|}{\left\| \vec{i} \right\|} \tag{3-11}$$

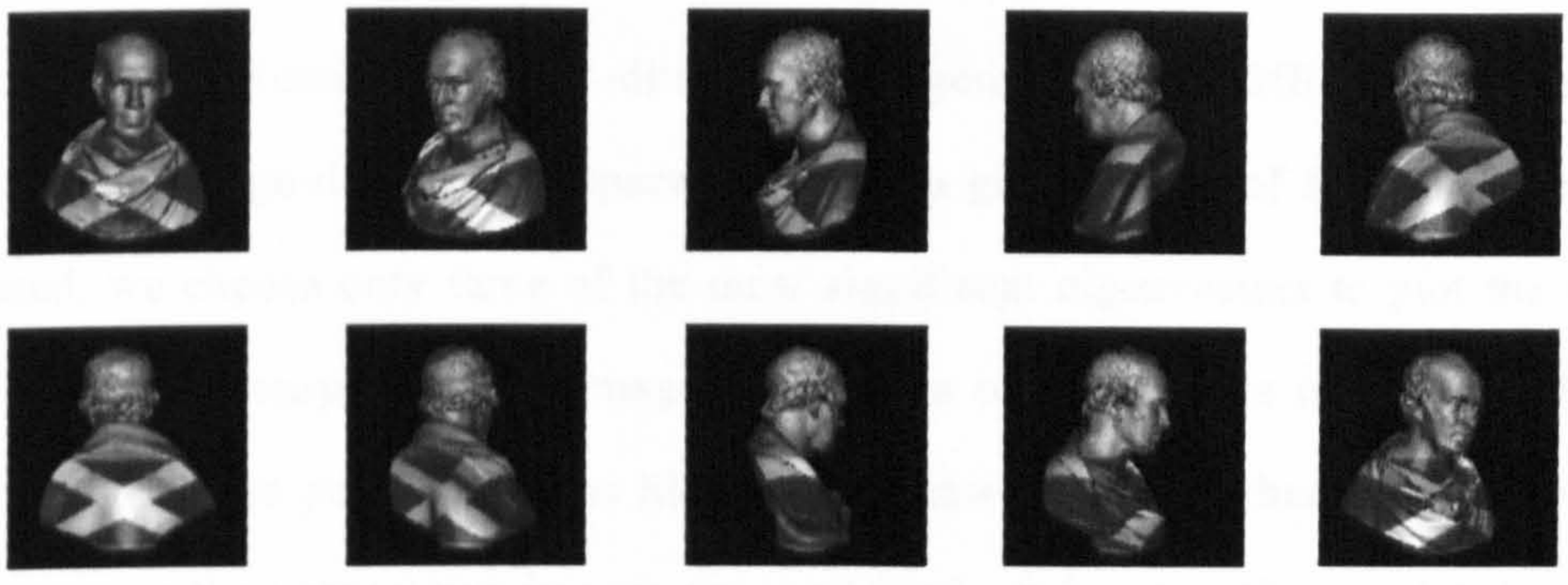


Figure 3-3 3D Eigenspace representation of the object

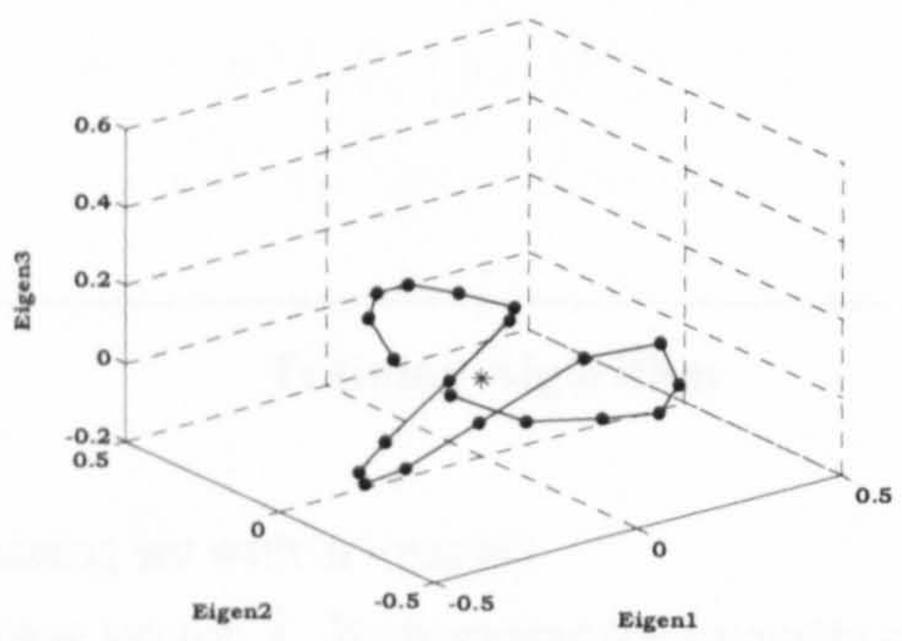


Figure 3-4 Ten images of a training set containing 20 images

where ||.|| denotes the Euclidean norm.

To compute the Eigenspace points of the training samples, we project the images of the training set into the Eigenspace. Each learning sample $\vec{\mathbf{i}}$ in the image set is projected to the Eigenspace by first subtracting the average image $\bar{\vec{\mathbf{i}}}$ and then finding the dot product of the result with each of the eigenvectors of the Eigenspace:

$$w_x = e_x^T \left(\vec{i} - \bar{\vec{i}} \right) \tag{3-12}$$

for x = 1, …, k. The weights form a vector

$$\Omega^T = [w_1, w_2, \cdots, w_k] \tag{3-13}$$

that describes the contribution of each eigenimage in representing the image, treating the eigenimages as a basis set for images.

By projecting all the images in the training set in this way, we get a set of discrete points, the model, in the Eigenspace. If we construct the Eigenspace with k eigenvectors, the

discrete points are distributed in a k-dimensional Eigenspace. It is difficult to display and visualize such a high-dimensional space. In order to give an idea of how the points are distributed, we choose only three of the most significant eigenvectors to plot the image points in a 3D Eigenspace. If the image set consists of consecutive images (see Figure 3-4), the Eigenspace points are most likely to be close to one another (see Figure 3-3). This is because the consecutive images are most likely to be strongly correlated, i.e. the more highly correlated are the images, the closer the projections are in Eigenspace. The training part of the algorithm is summarized below:

Training Algorithm

For (each image j in training set with n images)

- Form image vector $\vec{x}_j$ by scanning the intensity of image j from top to bottom and left to right.
- Form an energy normalized image vector $\vec{I}_j$ so that the total energy of one image is unity.

End

- Compute the mean image vector $\bar{\vec{I}}$ by $\bar{\vec{I}} = \left(\sum_{j=1}^{n} \vec{I}_j \right) / n$

For (each image vector $\vec{I}_j$)

- Compute the difference image vector $\vec{I}_j'$ by $\vec{I}_j' = \vec{I}_j - \bar{\vec{I}}$

End

- Compute the eigenvectors $\left[\vec{e}_1^{\,T}, \vec{e}_2^{\,T}, \cdots \vec{e}_k^{\,T} \right]$ and eigenvalues from the covariance matrix formed by $\left[\vec{I}_1'^{\,T}, \vec{I}_2'^{\,T}, \cdots \vec{I}_n'^{\,T} \right]$

For each $\vec{I}_j'$

- Compute the Eigenspace point coefficients $\vec{w}_j$ by projecting $\vec{I}_j'$ into Eigenspace $w_x = \vec{e}_x^{\,T} \vec{I}_j'$

End

3.1.4 Correlation and Euclidean distance in Eigenspace

Consider two images $\vec{i}_m$ and $\vec{i}_n$ that belong to the image set used to compute an Eigenspace. Let Ω_m and Ω_n be the Eigenspace projections of the two images. Then each image can be expressed as

$$\vec{i}_m = \sum_{i=1}^{N} w_i e_i + \bar{\vec{i}} \tag{3-14}$$

where $\bar{\vec{i}}$ is the mean image vector. The individual weights w_i are the coordinates of the point Ω_m. Since our Eigenspaces are composed of only K eigenvectors and these eigenvectors correspond to the largest eigenvalues, these represent the most significant variations within the image set. Hence, each image can be approximated as

$$\vec{i}_m \approx \sum_{i=1}^{N} w_{mi} e_i + \bar{\vec{i}} \tag{3-15}$$

where $K < N$. The similarity between two images can be determined by finding the correlation between brightness values in the images $\vec{i}_m{}^T \vec{i}_n$. The correlation between images is related to the sum-of-squared-differences (SSD) between brightness values in the images in that:

$$\left\| \vec{i}_m - \vec{i}_n \right\|^2 = \left(\vec{i}_m - \vec{i}_n \right)^T \left(\vec{i}_m - \vec{i}_n \right) \tag{3-16}$$

As the image brightness is normalized,

$$\left\| \vec{i}_m - \vec{i}_n \right\|^2 = 2 - 2\vec{i}_m{}^T \vec{i}_n \tag{3-17}$$

Thus, maximizing the similarity between the images is equivalent to maximizing the correlation between them, which in turn corresponds to minimizing the SSD. The SSD can be expressed in terms of the Eigenspace points Ω_m and Ω_n:

$$\begin{aligned} \left\| \vec{i}_m - \vec{i}_n \right\|^2 &\approx \left\| \sum_{i=1}^{K} w_{mi} e_i - \sum_{i=1}^{K} w_{ni} e_i \right\|^2 \\ &\approx \left\| \sum_{i=1}^{K} (w_{mi} - w_{ni}) e_i \right\|^2 \\ &\approx \sum_{i=1}^{k} \sum_{j=1}^{K} e_i{}^T e_j (w_{mi} - w_{ni})^2 \end{aligned} \tag{3-18}$$

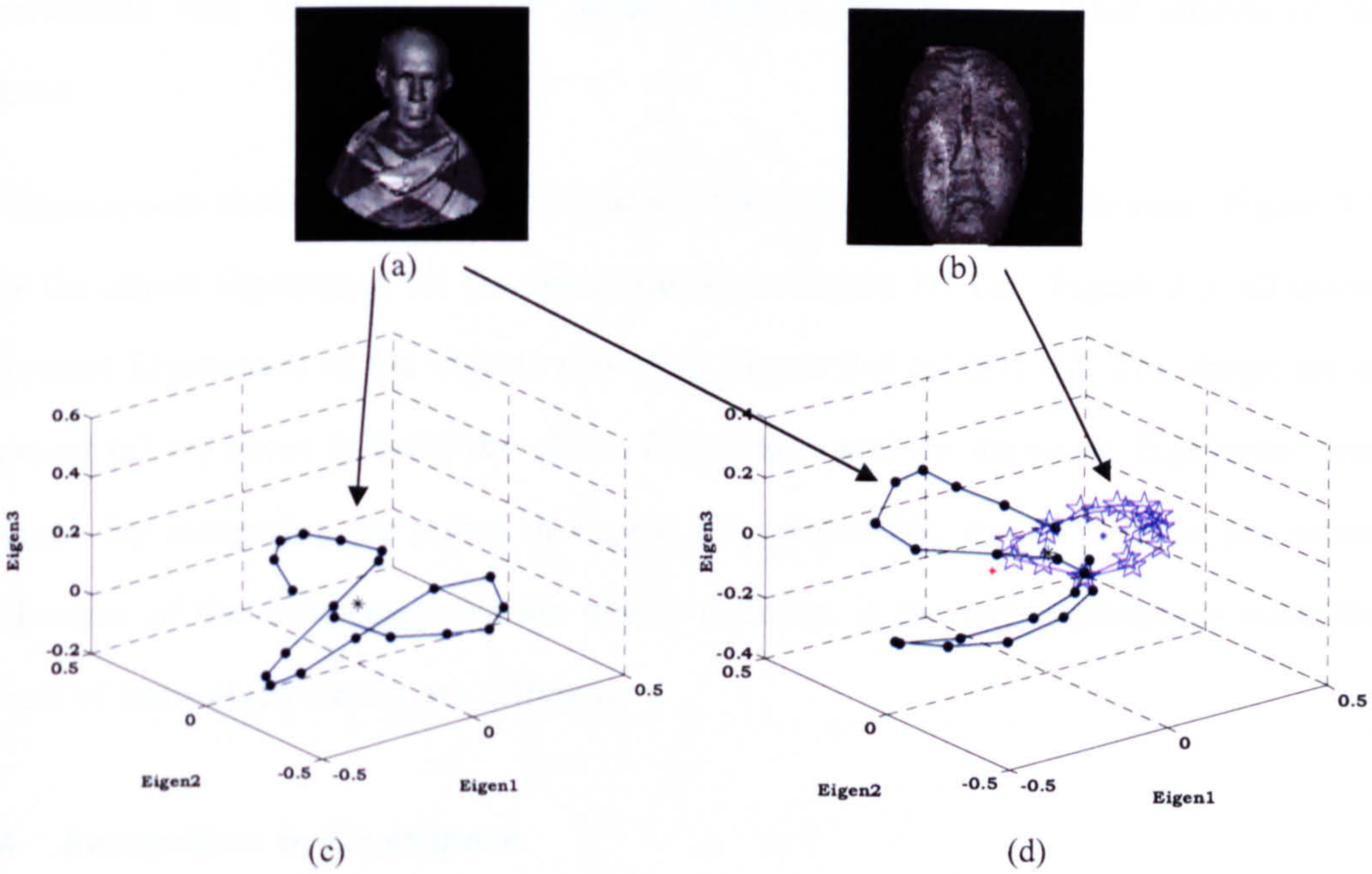


Figure 3-5 Demonstration of universal and object Eigenspaces. (c) is the object Eigenspace for object (a), (d) is the universal Eigenspace for objects (a) and (b).

Since the eigenvectors are orthonormal: $e_i^T e_j = 1$ when $i = j$, and 0 otherwise, we get:

$$\left\| \vec{i}_m - \vec{i}_n \right\|^2 \approx \left\| \Omega_m - \Omega_n \right\|^2 \tag{3-19}$$

The above relation implies that the square of Euclidean distance between points Ω_m and Ω_n is an approximation to the SSD between images $\vec{i}_m$ and $\vec{i}_n$. In other words, the closer the projections are in Eigenspace, the more highly correlated are the images.

3.1.5 Universal Eigenspace and object Eigenspace

We have implemented two ways of building the Eigenspace: building the universal Eigenspace and building the object Eigenspace. The universal Eigenspace is computed using the image set of all objects of interest to the recognition system. The object Eigenspace is computed using only images of one object. For example, if there are n objects in the database, n object Eigenspaces are needed for the recognition system. The universal Eigenspace is computationally exhaustive. The set of object Eigenspaces is flexible in that it can be easily expanded as each new object requires the computation to

be performed only an image of that object, without reference to other objects in the database.

The Eigenspaces shown in Figure 3-5 illustrate this principle in a simple case. Figure 3-5 (c) is the object Eigenspace for the object shown in Figure 3-5 (a) . Figure 3-5 (d) shows a universal Eigenspace of the objects shown in Figure 3-5 (a) and (b). The image set of the object (a) we chose to build the object Eigenspace and the universal Eigenspace was the same. By comparing the points in Figure 3-5 (c) and (d), we see that the projection distributions of the same image set are totally different in the two Eigenspaces since the two sets of basis eigenvectors are different.

3.1.6 Recognition in Eigenspace

To identify an input image, we project it to a certain Eigenspace to get a weight vector of that input image. The vector may then be used in a standard pattern recognition algorithm to find which of a number of predefined object classes, if any, best describes the object in the input image.

The similarity between the two images can be determined by finding the sum-of-squared-difference (SSD) between brightness values in the images. Unfortunately, direct application of the SSD is rather expensive. We have to therefore develop a simpler, more efficient method. The distance between image vectors can be approximated by the distance in the k-dimensional Eigenspace (Equation (3-19)). In other words, the closer the projections are in Eigenspace, the more highly correlated are the images. In the recognition module, we consider an image of a scene that includes one of the objects we have learned. We assume that the image regions corresponding to the object have been separated from the scene image. Each segmented image region is normalized in scale and energy as described in Section 3.1.4. This ensures that:

- the $m \times n$ input image has the same number of elements as the [$m \times n$ by 1] eigenvectors of the universal and individual Eigenspaces;

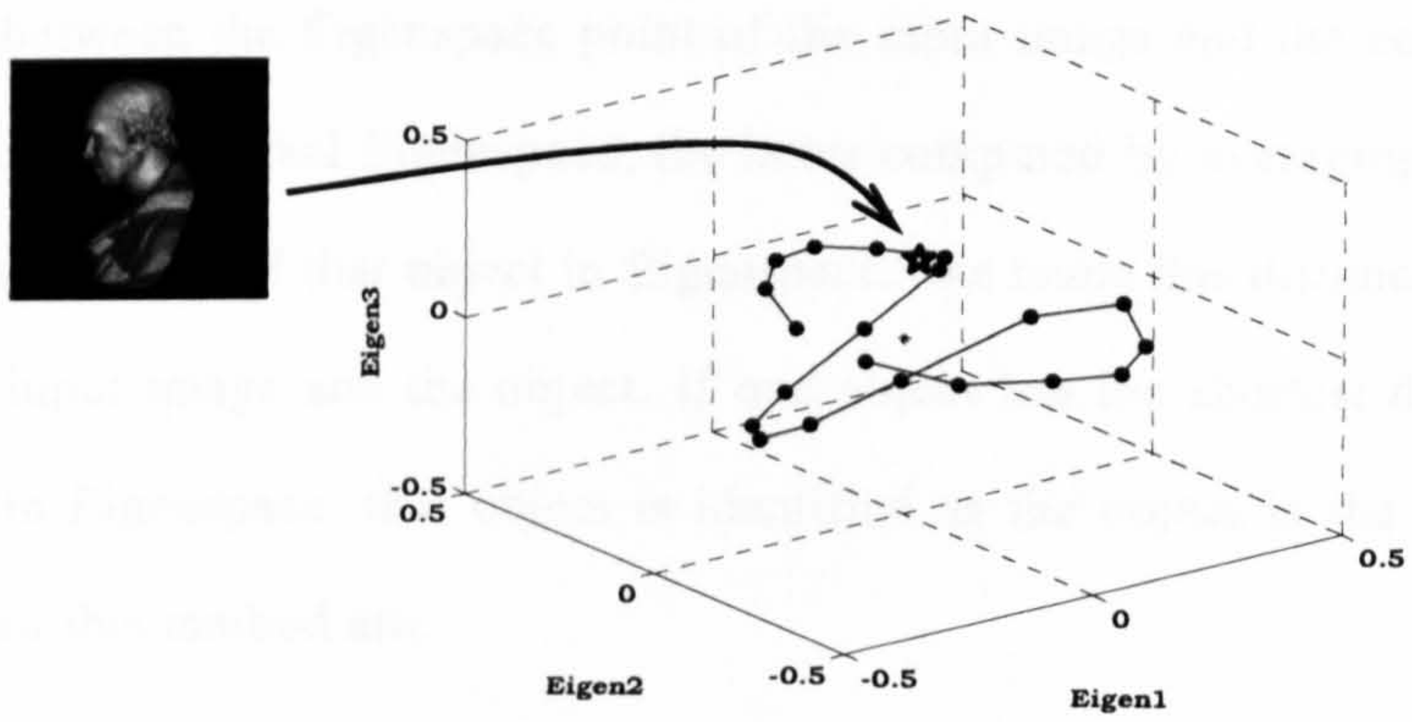


Figure 3-6 An input image and its projection in Eigenspace

- the recognition system is invariant to fluctuation in the global intensity of illumination;
- the recognition system is invariant to magnification, i.e. the distance of objects from the sensor.

Computing the Eigenspace Point of the Input Image

Before projecting the input image to the Eigenspace, we need to subtract the mean image, produced in building that Eigenspace.

Then we do the inner product of the input image and the eigenvectors to get the weight vector. Figure 3-6 shows an input image of the object whose Eigenspace was shown in Figure 3-3. The input image we choose is not the same as any of the images in the training set, but an image between pose 5 and pose 6 of the object in the training set. In Figure 3-6, the round points shows the projection of the points in the training set and the five-point star represents the projection of the input image. We see that the position of the input image is nearly on a straight line between pose 5 and pose 6 of the training set.

Identifying the Object in the Input Image

There are two ways of identification: comparative identification and categorical identification. Comparative identification is used in a universal Eigenspace. We compare

the distance between the Eigenspace point of the input image and the central location of each object in the universal Eigenspace, the latter computed by averaging the coordinates of all the image points of that object in Eigenspace. We name this distance as the distance between the input image and the object. If one object has the shortest distance with the input image in Eigenspace, that object is identified as the object in the image. The two assumptions of this method are:

(i) The mean locations of different objects are well separated.

(ii) The Eigenspace points of images of one object are relatively close.

Categorical identification can be used in either a universal Eigenspace or individual Eigenspace. The key idea of this method is to establish a threshold. If the distance between the input image and one training sample in the Eigenspace is below the threshold, we judge that the input image and the training sample represent the same object.

3.2 Interpolation of the Object Manifold in Eigenspace

In section 3.1.1, we say that the appearance model is parameterized by the image acquisition variables. In section 3.1.2 and 3.1.3, we build the Eigenspace and project all the training images to the Eigenspace. Until then, the appearance model is represented by the Eigenspace and the discrete points formed from the training set. In this section, we aim to build a more advanced appearance model by finding the relationship between the training image points and the image acquisition variables.

3.2.1 Parametric Eigenspace

Since images with consecutive acquisition variables are strongly correlated, their projections in Eigenspace are close to one another[4] (see section 3.1.4). The discrete points

[4] In the cases when the object is either highly specular or has high frequency, an incremental pose can cause dramatic changes in image brightness. However, the objects in our database does not have such an effect.

formed from the training set can be considered as samples from a smoothly varying manifold in Eigenspace (see Figure 3-3) parameterised by image acquisition variables:

$$\Omega = g(q) \tag{3-20}$$

where Ω is a matrix in which each column is the coefficient vector of each training image point, q is a collection of discrete image acquisition variables (see Equation (3-1)), and g is the presentation of the manifold we aim to discover, referred to as parametric Eigenspace representation. Depending on the variability of the training set, the appearance representation may be a curve or a surface in a k-dimensional space. If q represents one variable, Equation (3-20) is a parametric form of a multidimensional curve; if q represents two variables, Equation (3-20) is a parametric form of multidimensional surface. The dimensionality equals to the number of Eigenvectors used. For example, if we use three Eigenvectors and we have two variables, Equation (3-20) can be written as

$$\begin{aligned} \Omega_x &= g_x(q_1, q_2) \\ \Omega_y &= g_y(q_1, q_2) \\ \Omega_z &= g_z(q_1, q_2) \end{aligned} \tag{3-21}$$

where the three equations are independent of each other. The idea of a parametric Eigenspace representation is mentioned firstly in Murase and Nayar's work [68].

The parametric Eigenspace is a compact representation of the appearance of an object. With this representation, we can predict the Eigenspace points of new images with acquisition variables' value in between any training images, e.g., if we have training images at pose of 30 degrees and 40 degrees in azimuth angle, we could predict the image point of the image from pose 35 degree with a certain accuracy. Furthermore, the recognition rate is improved when any new image of the object is presented. Now we have a set of sample points in Eigenspace which lie on a smooth manifold parameterised by some image acquisition variables and we are interested to know some other points in between those known points. This can be solved by interpolation. In the next section, we will discuss the use of interpolation in detail and evaluate its effect.

3.2.2 Interpolation and its Evaluation

Interpolation is a method of constructing new data points from a discrete set of known data points. It is distinct from fitting a function to a series of points. In particular, an interpolated function goes through all the original points while a fitted function may not. The interpolation function should approximate the original function to as high a degree as possible. For example, assume that the original function has several orders of continuous derivatives so that Taylor's theorem applies. Thus the interpolation function and the Taylor series expansion for original function should agree for as many terms as possible. As a reminder, the Taylor series of a function $f(x)$ that is infinitely differentiable in the neighbourhood of a number a is,

$$f(x) = f(a) + \frac{f'(a)}{1!}(x-a) + \frac{f^2(a)}{2!}(x-a)^2 + \cdots + \frac{f^n(a)}{n!}(x-a)^n + Rn \qquad (3\text{-}22)$$

A second-order Taylor series expansion of a scalar-valued function of more than one variable can be compactly written as

$$T(x) = f(a) + Df(a)^T(x-a) + \frac{1}{2!}(x-a)^T D^2 f(a)(x-a) + \cdots \qquad (3\text{-}23)$$

where $Df(a)$ is the gradient, and $D^2 f(a)$ is the Hessian matrix:

$$H(f) = \begin{bmatrix} \frac{\partial^2 f}{\partial x_1^2} & \frac{\partial^2 f}{\partial x_1 \partial x_2} & \cdots & \frac{\partial^2 f}{\partial x_1 \partial x_n} \\ \frac{\partial^2 f}{\partial x_2 \partial x_1} & \frac{\partial^2 f}{\partial x_2^2} & \cdots & \frac{\partial^2 f}{\partial x_2 \partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2 f}{\partial x_n \partial x_1} & \frac{\partial^2 f}{\partial x_n \partial x_2} & \cdots & \frac{\partial^2 f}{\partial x_n^2} \end{bmatrix} \qquad (3\text{-}24)$$

In Equation (3-20), Ω contains the coordinates of the series of points, g is the targeting function and q is the direction of the interpolation. If q represents one variable, the interpolation is one dimensional; if q represents two or more variables, the interpolation is two-dimensional or multi-dimensional. In the coil-100 database, each object is represented by 72 images at pose (azimuthally) intervals of 5 degree. In this case, q contains one image acquisition variable, pose in longitude, having the values of 5, 10, 15, ..., 355 in degree, or 1, 2, 3, ... 72 in pose index. The fact that q is defined in only

one variable means that the data is sampled along one dimension. Moreover, if each dimension is independent of each other, then although Ω contains multidimensional points, the problem is still a one dimensional interpolation problem. Each dimension can be interpolated separately. In the following text, we only work with the first dimension in Eigenspace Ω_x, and the same procedure can be expanded to multidimensions. In the image acquisition procedure, if we move the camera position along both longitude and latitude, we have two image acquisition parameters. In this case, q represents two variables, pose in longitude and pose in latitude. The corresponding interpolation will be two-dimensional. In the case when the other additional imaging conditions are changed, e.g., lighting or thermal conditions, the interpolation will be multi-dimensional.

1D-Interpolation

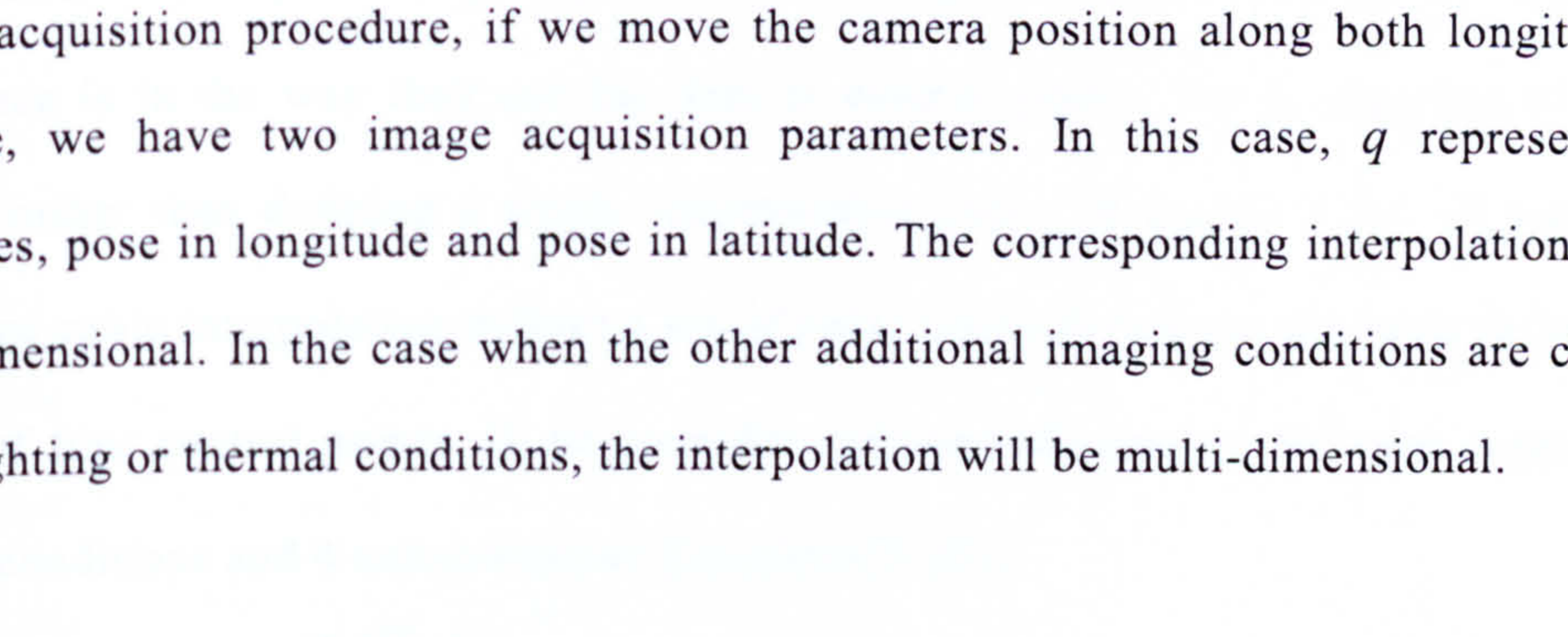


Figure 3-7 Linear Interpolation

The simplest interpolation is linear interpolation in one dimension. For example, Figure 3-7 illustrates linear interpolation of the function $f(x)=\sin(\pi x)$ over the range $-1 \le x \le 1$, which covers eight uniformly-sized data sampling intervals. We see that the method fits a linear function at each interval. The interpolation error is zero at sampling points but not within each sampling interval. The linear interpolation is the simplest polynomial interpolation. In its most general form a polynomial interpolation consists of polynomial pieces at each sampling intervals. For example, in linear interpolation, the

interval functions are of degree 1. The linear interpolation is a entirely local method and as a result the function is not smooth.

Cubic interpolation of a curve is used more commonly than a linear method. There are various types of cubic interpolation. Here we consider Lagrange cubic interpolation [97], cubic Hermite interpolation [98] [99] and cubic spline interpolation [100]. The major difference is in the way they use the data at control points. For a sequence of control points, rather than defining a single interpolation curve of degree 3 for all points, the Lagrange cubic interpolation defines a set of cubic interpolating curves, each defined by a group of four control points. If we consider only one dimension, for each segment, we have 4 conditions and 4 unknowns(see Equation (3-25)).

$$\begin{aligned} g_x(0) &= c_0 \\ g_x(1) &= c_0 + c_1 + c_2 + c_3 \\ g_x(2) &= c_0 + 2c_1 + 2^2 c_2 + 2^3 c_3 \\ g_x(3) &= c_0 + 3c_1 + 3^2 c_2 + 3^3 c_3 \end{aligned} \tag{3-25}$$

By solving the linear equations, for each segment, we could express $g_x(q)$ in terms of blending polynomials as

$$g_x(q) = \sum_{i=0}^{3} b_i(q)\Omega_{xi} \tag{3-26}$$

where the Ω_{xi} is the x-coordinates of points 0 to 3, $b_i(q)$ is the blending functions (see Figure 3-8). The continuity at the joining points is achieved by using the control point defining the right side of one segment as the first point for the next segment. The Lagrange cubic interpolation is a straightforward approach in that it uses four points to solve for the four unknown coefficients. However, the segmentation of continuous points into groups of four does not make sense in our application.

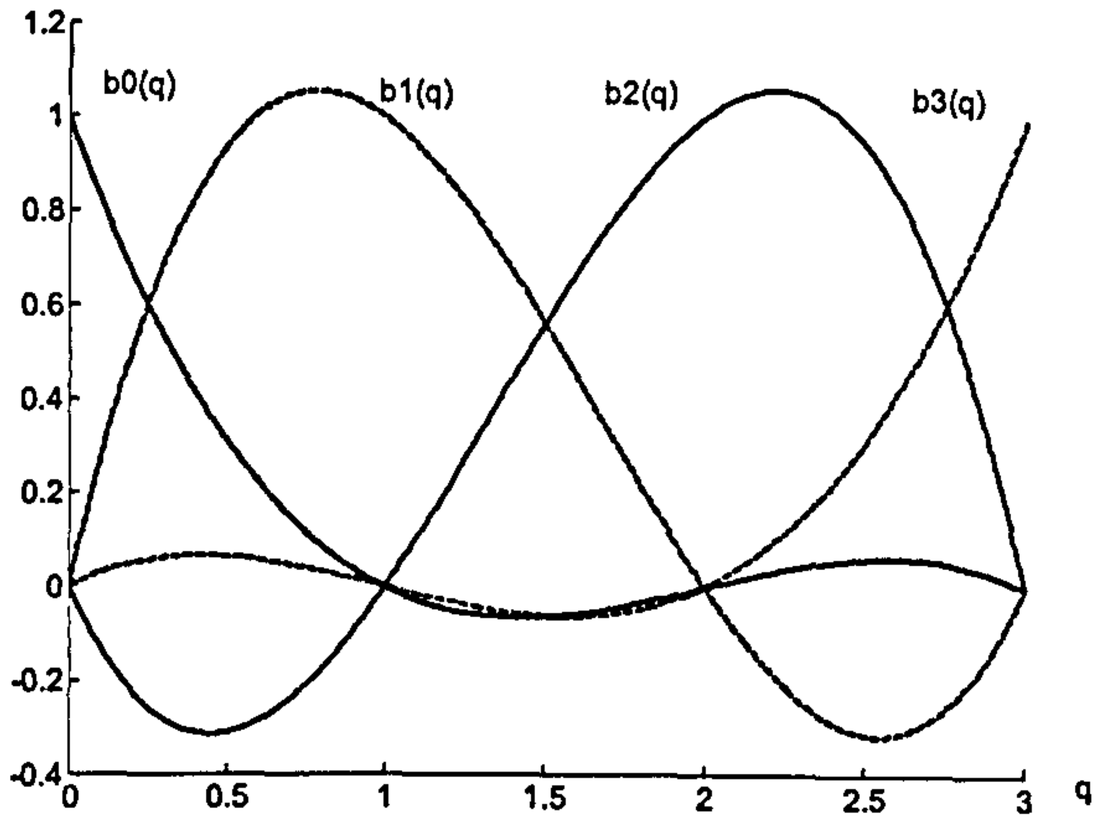


Figure 3-8 Blending polynomials for Lagrange cubic interpolation

The cubic Hermite interpolation is also a local interpolation. However, each polynomial of the interpolation is in a Hermite form that consists of two control points and two control tangents for each polynomial (see (3-27)).

$$\begin{aligned} g_x(0) &= c_0 \\ g_x(1) &= c_0 + c_1 + c_2 + c_3 \\ g_x'(0) &= c_1 \\ g_x'(1) &= c_1 + 2c_2 + 3c_3 \end{aligned} \tag{3-27}$$

By solving the linear equations, for each pair, we could express $g_x(q)$ in terms of blending polynomials as

$$g_x(q) = B^T \Omega_x' \tag{3-28}$$

where the first two elements of Ω_x' are the coefficients of the two control points and the other two are the slopes at the control points, and B are the blending functions as shown in Figure 3-9. The slopes at the $q(j)$ are chosen in such a way that $g(q)$ is "shape preserving" and "respects monotonicity". When the required slopes at the interpolation points is not available, it is a simple matter to use local data to construct an approximation to the slope at each point in turn, e.g., we could define a slope at a point by taking the slope of the line through its nearest two points. On each subinterval, $q(k) \le q \le q(k+1)$, $g(q)$ is the cubic Hermite interpolant to the given values and certain slopes at the two endpoints. Therefore, $g(q)$ interpolates Ω, i.e., $g(q(j)) = \Omega(j)$, and the first derivative, $g'(q)$, is continuous, but $g''(q)$ is probably not continuous; there may be jumps at the $q(j)$. In cubic Hermite interpolation, the subdomain over which local interpolation can

be done comprises the closed interval defined by two consecutive data points. The interpolant within one subregion will not affect, nor be affected by, the data in any other subregion.

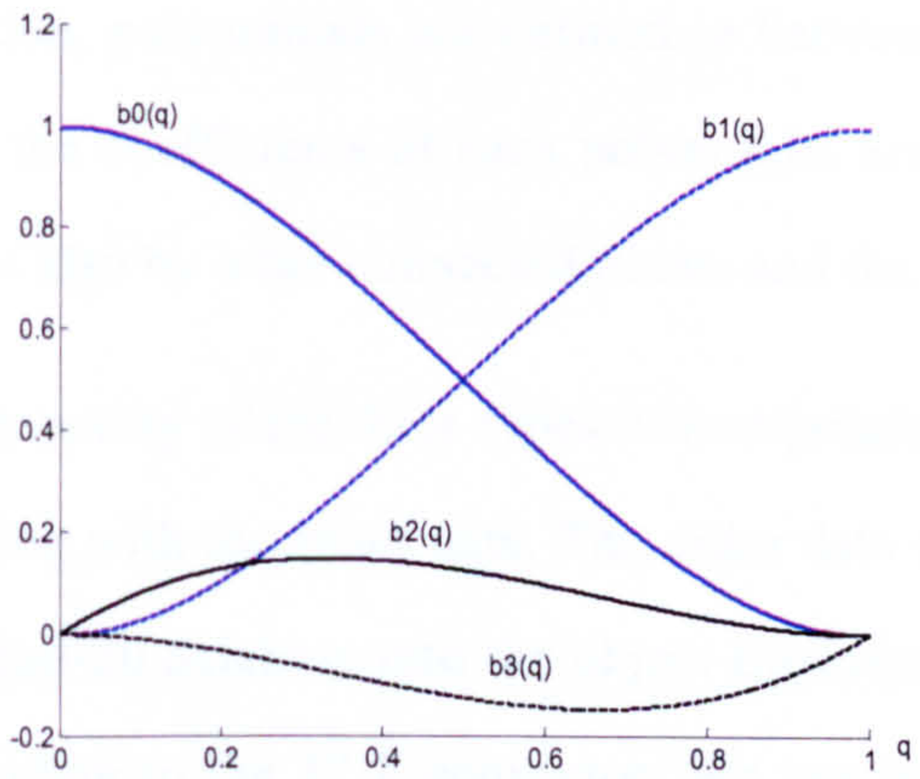


Figure 3-9 The cubic Hermite basis

The cubic spline interpolation fits between each pair of existing data points a different cubic function of the form:

$$g_x(q) = \begin{cases} g_{1x}(q) & q \in [q_0, q_1] \\ g_{2x}(q) & q \in [q_1, q_2] \\ \vdots & \\ g_{(n-1)x}(q) & q \in [q_{n-1}, q_n] \end{cases} \tag{3-29}$$

The difference from cubic Hermite interpolation is that the slopes at the $q(j)$ are chosen differently, namely to make even $g''(q)$ continuous. Suppose we have n intervals and four coefficients for each to determine 3rd degree polynomials at each interval. We require a total of $4n$ parameters. For each interval, we require that the polynomial pass the two ends and that gives $2n$ parameters. For the joint of each two intervals, we require the first and second derivatives to be continuous and that gives $2(n-1)$ parameters. The last two conditions can be defined at the end point, called end conditions. For natural cubic splines, the end condition is that the $g''(q)$ are zeros at the end points. Then the interpolation at each interval can be defined as:

$$g_{ix}(q) = \frac{a_{i+1}(q-q_i)^3 + a_i(q_{i+1}-q)^3}{6h_i} + \left(\frac{\Omega_{x(i+1)}}{h_i} - \frac{h_i}{6}a_{i+1}\right)(q-q_i) + \left(\frac{\Omega_{xi}}{h_i} - \frac{h_i}{6}a_i\right)(q_{i+1}-q) \tag{3-30}$$

where $h_i = q_{i+1} - q_i$ and the coefficients can be found by solving this system of equations:

$$\begin{cases} a_0 = 0 \\ h_{i-1}a_{i-1} + 2(h_{i-1} + h_i)a_i + h_i a_{i+1} = 6\left((\Omega_{x(i+1)} - \Omega_{xi})/h_i - (\Omega_{xi} - \Omega_{x(i-1)})/h_i\right) \\ a_n = 0 \end{cases} \quad (3\text{-}31)$$

In cubic spline interpolation, polynomials are defined in between each connected pair of control points. However, the coefficients of each polynomial are not only determined by the two control points, but also by other connected points and the end conditions.

Figure 3-10 illustrates the quality of the three types of interpolation, linear, cubic Hermite, and cubic spline, comparing with the exact data. The exact data is obtained by projecting the 72 cat images in the coil-20 database into the object Eigenspace and considering only the coefficients corresponding to the 1st Eigenvector. We use poses at intervals of four, that is, pose 1, 5, 9, 13, ..., 69, 1(73), as the 19 sampling points, and predict the three points at each interval using the interpolation methods. From Figure 3-10, we see that the interpolation result from the cubic Hermite method is better than that of the linear method, e.g., in intervals between sampling point 5 to 7, 8 to 9, 10 to 11 etc. The data interpolated by the cubic spline has the smallest error compared with exact data among these three methods. Since the coefficients in Eigenspace are independent from each other and each coefficient is a function of the pose parameter, they can be interpolated using the same method.

Then we evaluate the cubic spline interpolation using different numbers of samples. We build an Eigenspace using all 72 images of the 'duck' (see the up-left image in Figure 3-47 for an image of the 'duck') in the Coil-20 database (the 72 images are got by fixing the object and moving the camera clockwise at 5 degrees interval) and to get 72 points in the multidimensional Eigenspace. We choose half of the points (pose 1, 3, 5, ..., 71) as training samples and use the spline interpolation to predict the rest. In this case, pose 1, 3, 5, ..., 71 provide the knots of the spline. Figure 3-11 shows a comparison between the predicted data and the true data. Here, the coefficient along each direction is interpolated separately. We can see that the spline interpolation performs well when predicting one

point in between each pair of knots. Then we reduce the number of samples and try to use the samples to predict more data, e.g., we use 24 samples to predict 46 data points, 18 samples to predict 51 data points, 15 samples to predict 56 data points (see Figure 3-12, Figure 3-13 and Figure 3-14). From Figure 3-11 to Figure 3-14, we can see some properties of the spline interpolation used in our application:

(i) More sample points give better prediction;

(ii) The resultant spline passes through the sample points;

(iii) For cases when not very many samples are known, if the curve between each pair of samples is smooth and of lower degree, the prediction can be accurate.

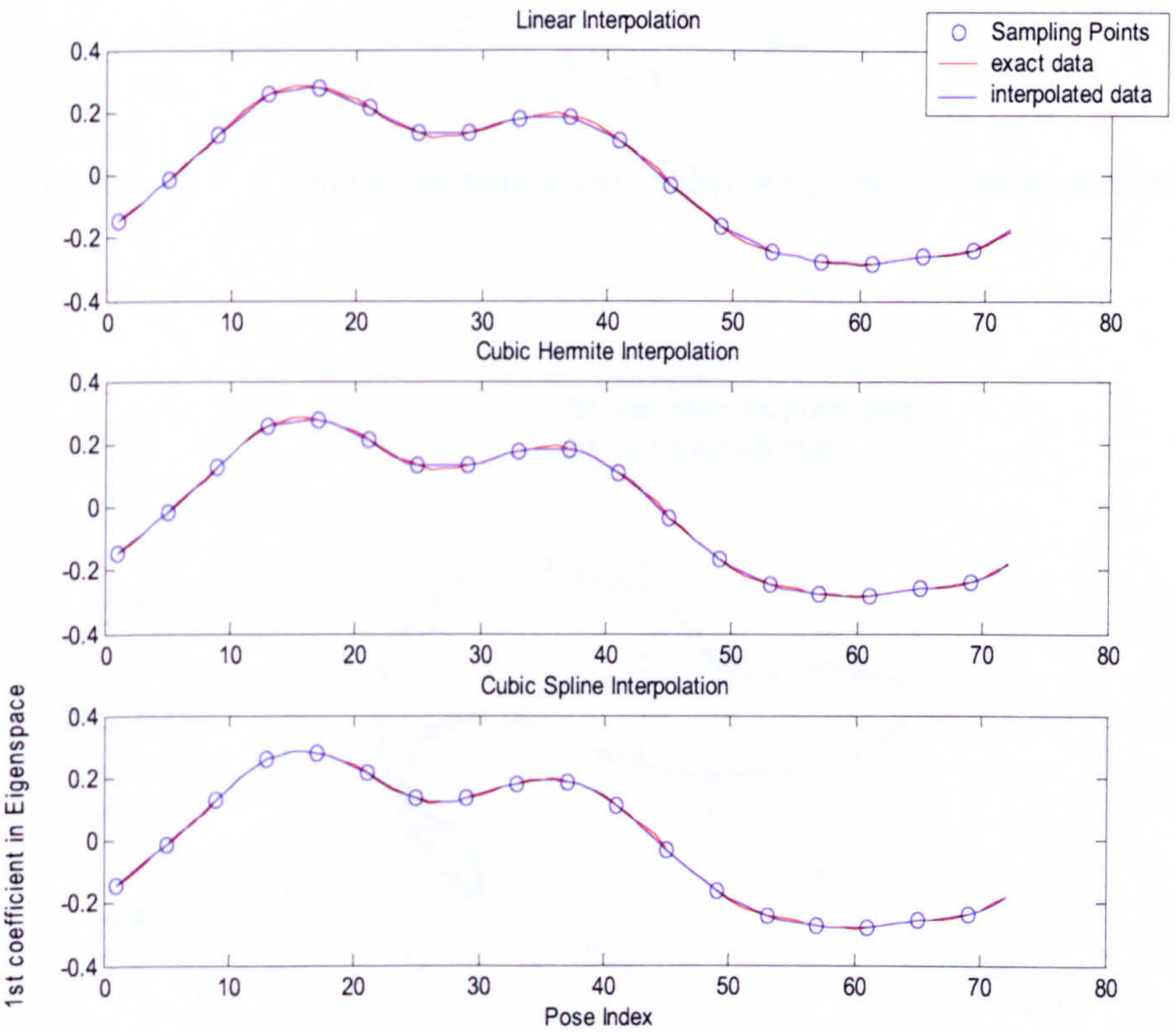


Figure 3-10 Compare the three interpolation using obj4 in Coil-20

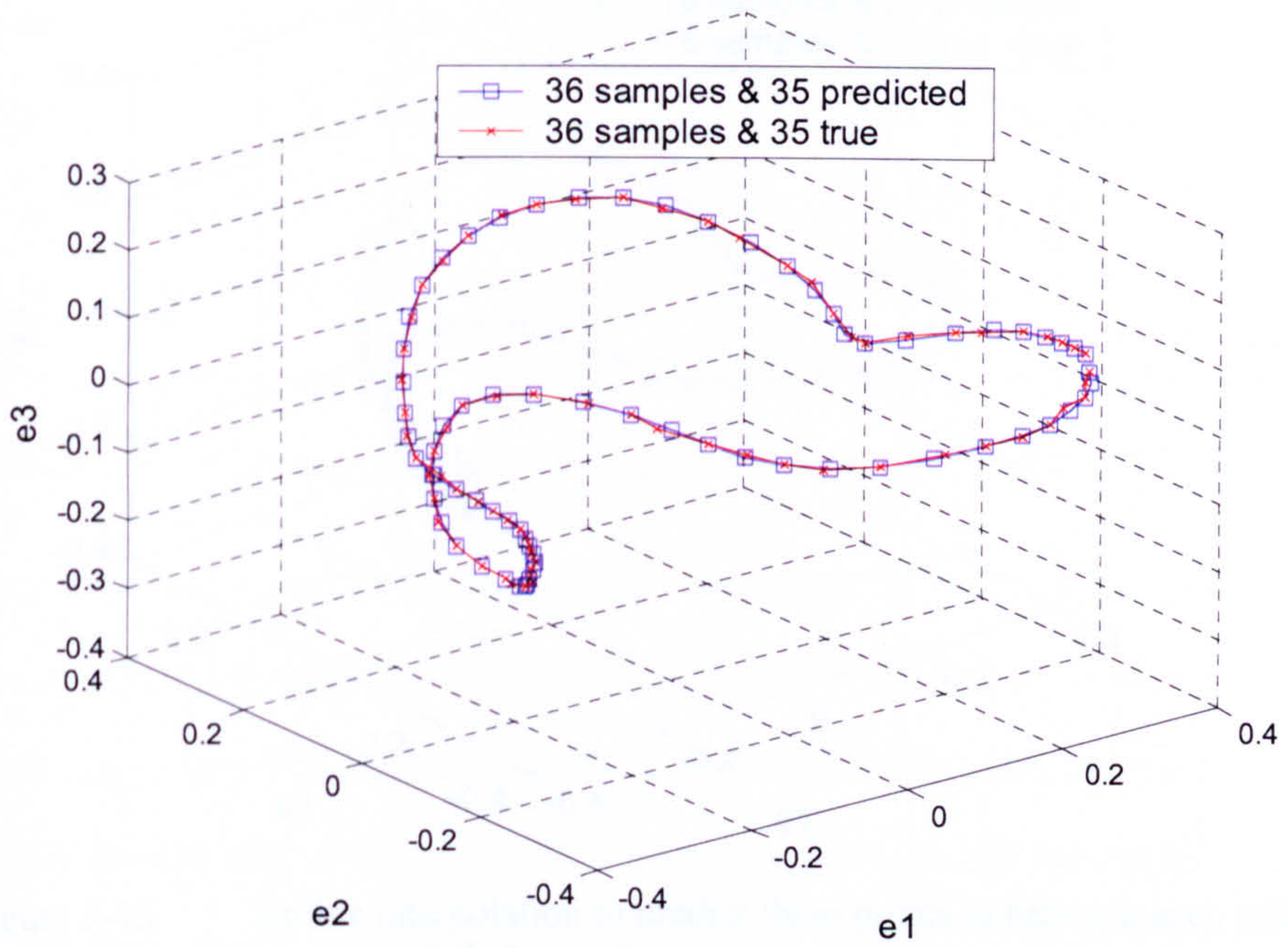


Figure 3-11 Spline interpolation to predict one point in between each pair of knots

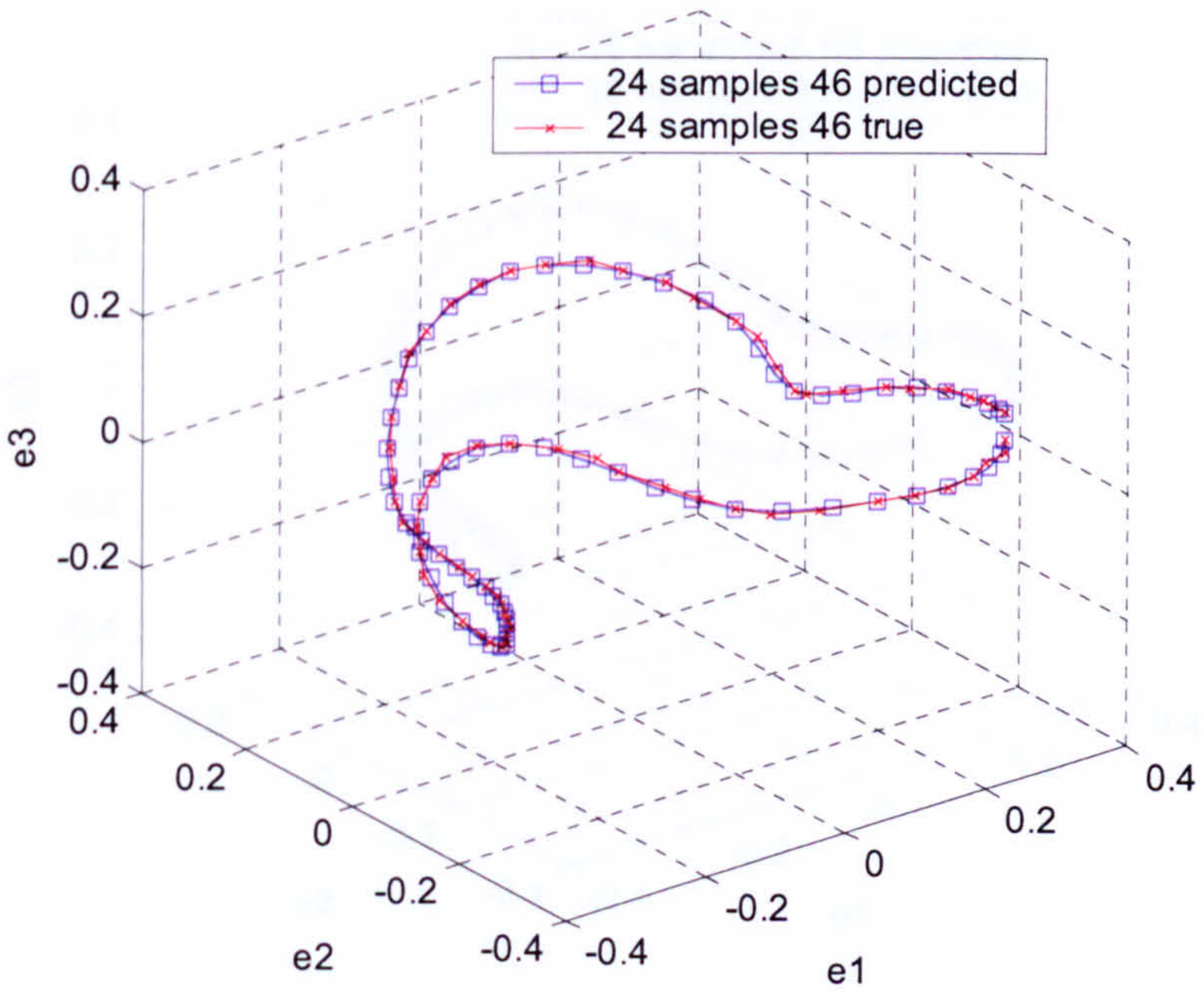


Figure 3-12 Spline interpolation to predict two points in between each pair of knots, the sample poses are 1, 4, 7, ..., 70.

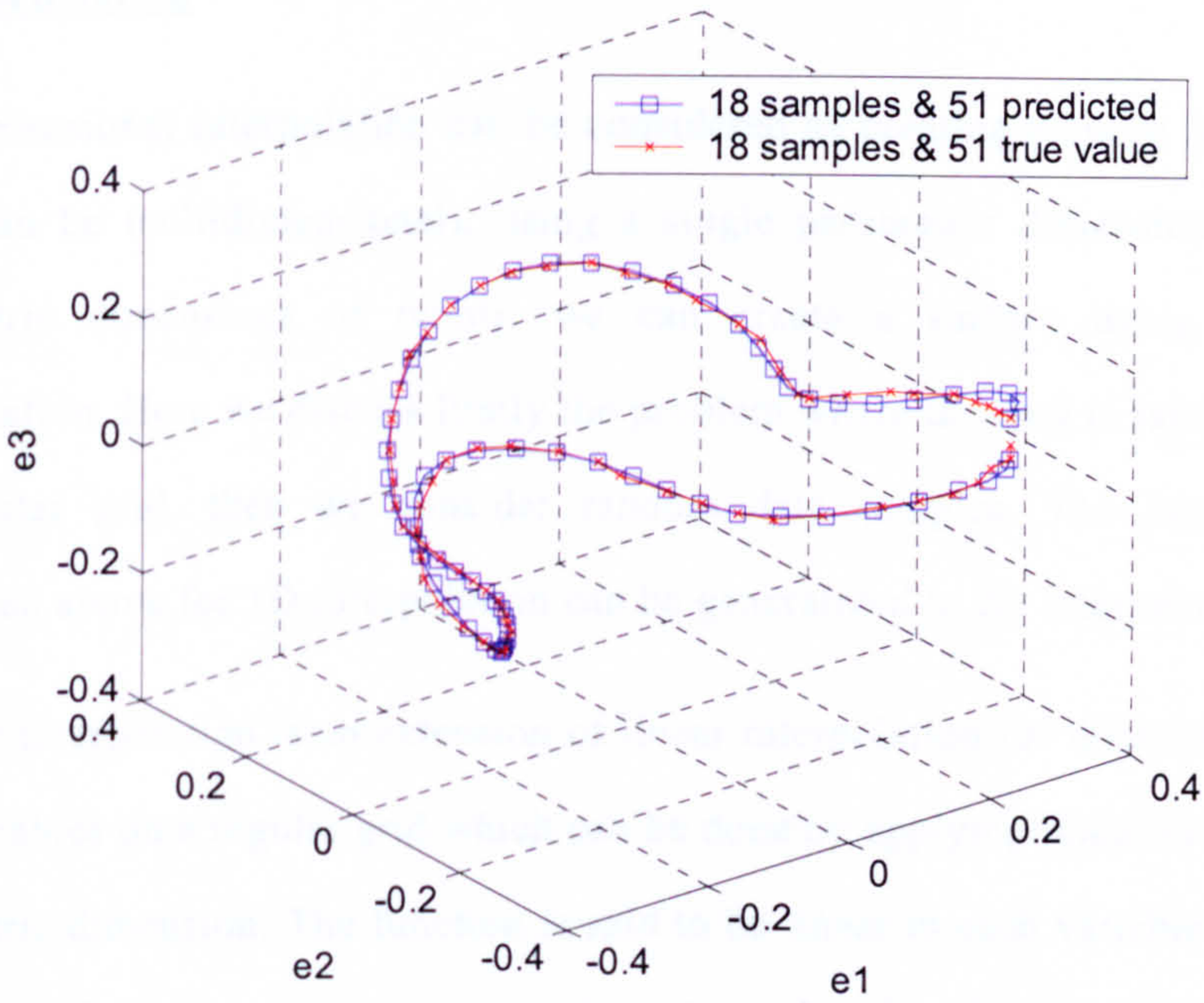


Figure 3-13 Spline interpolation to predict three points in between each pair of knots, the sample poses are 1, 5, 9, ..., 69.

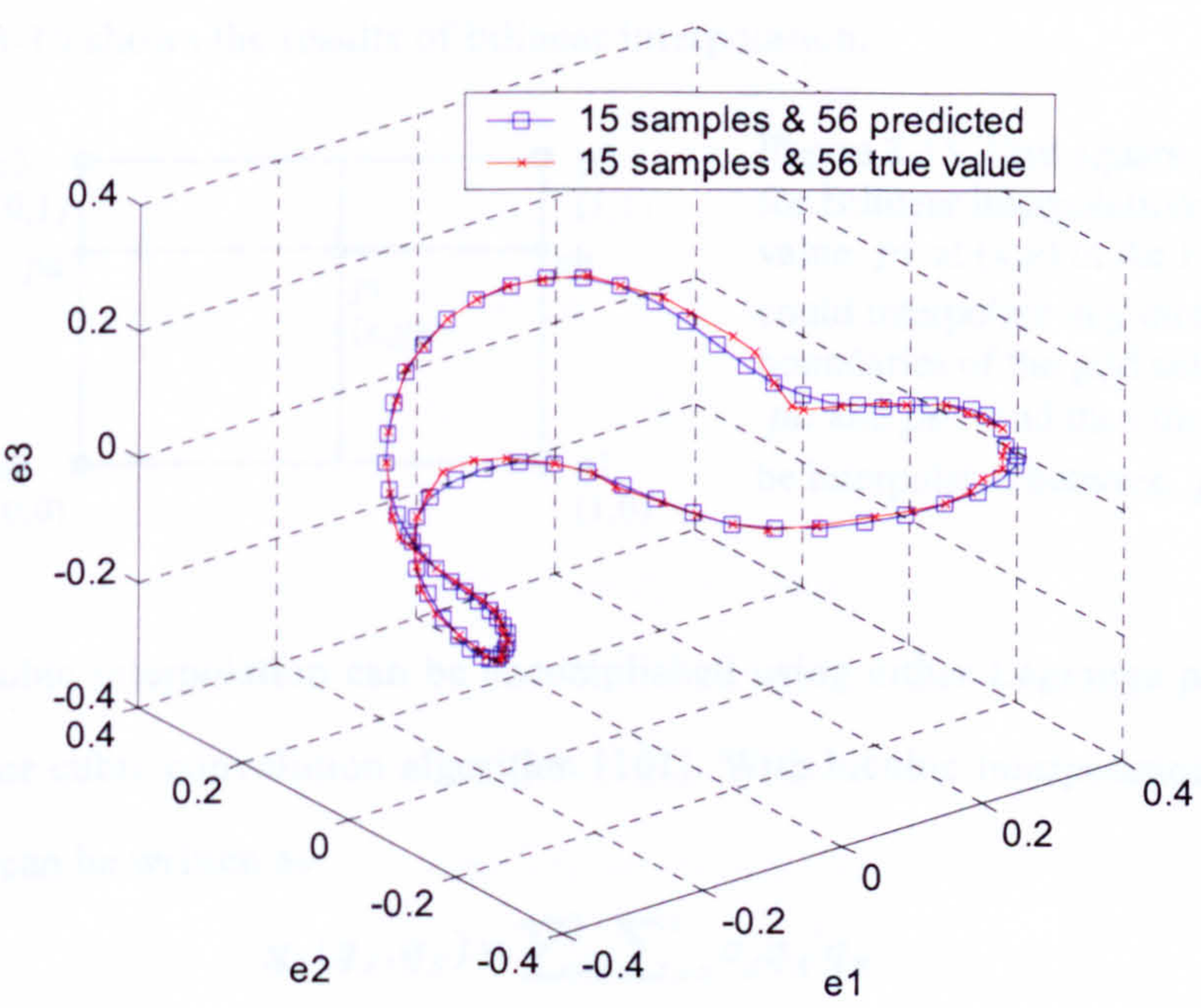


Figure 3-14 Spline interpolation to predict four points in between each pair of knots, the sample poses are 1, 6, 11, ..., 71.

2D-Interpolation

One dimensional interpolation can be considered as creating a curve (although the curve itself can be multidimensional), using a single parametric dimension. If we have two parametric dimensions of points, we can create a surface using two dimensional interpolation. Here we discuss firstly the problem where the data is sampled on a uniform, rectangular grid, then we consider random data samples. The interpolation method illustrated above for 1D interpolation can be generalized to 2D interpolation.

Bilinear interpolation is an extension of linear interpolation for interpolating functions of two variables on a regular grid which can be done by applying linear interpolation to each parametric dimension. The function is said to be linear in each variable when the other is held fixed. For example, to determine the value pi at (x, y)in the Figure below, we could interpolate in y direction at the boundaries of the grid cell to get pa and pb. And then the x coordinate can be interpolated between pa and pb.

Figure 3-16 shows the results of bilinear interpolation.

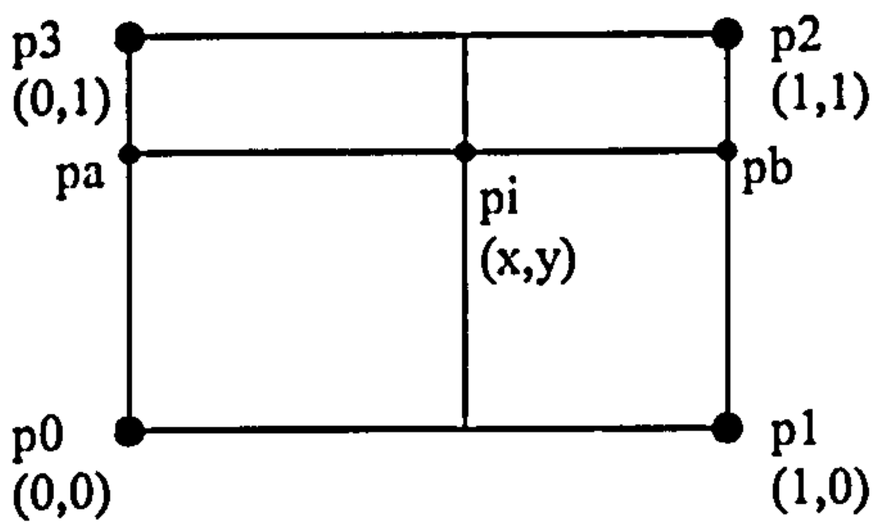


Figure 3-15 Unit square grid cell layout for Bilinear interpolation: to determine the value pi at (x,y) in the Figure below, we could interpolate in y direction at the boundaries of the grid cell to get pa and pb. And then the x coordinate can be interpolated between pa and pb.

The bicubic interpolation can be accomplished using either Lagrange polynomials, cubic splines or cubic convolution algorithm [101]. With bicubic interpolation, the interpolated surface can be written as

$$g_x(q_A, q_B) = \sum_{i=0}^{3} \sum_{j=0}^{3} a_{ij} q_A{}^i q_B{}^j \tag{3-32}$$

The interpolation problem consists of determining the 16 coefficients a_{ij}. The Lagrange polynomials use 16 control points to define an interpolating surface patch to get 16 equations in 16 unknowns. It is proven that the surface patch can be rewritten as

$$g_x(q_A,q_B)=\sum_{i=0}^{3}\sum_{j=0}^{3}b_i(q_A)b_j(q_B)\Omega_{xij} \tag{3-33}$$

where $b_i(q_A)$ and $b_j(q_B)$ are the same blending functions as in Equation (3-26), see Figure 3-8.

The bicubic spline interpolation, however, uses only 4 control points, together with three derivatives at each control point, the first derivatives expressing the slope of the surface in direction q_A and direction q_B, and the second (cross) derivative representing the slope in both q_A and q_B. The bicubic spline interpolation does not only match g and its first-order derivatives $\partial g/\partial q_A$ and $\partial g/\partial q_B$ at the data sampling points, but also matches all mixed first-order derivatives, $\partial^2 g/\partial q_A \partial q_B$. For each patch, the local coordinates and the estimated slopes can be input into the equations to generate 16 equations to solve the problem. The lower order derivatives approximation can be found in many texts. The better the approximation, the better the performance of the interpolation. Bicubic spline interpolation is the lowest order 2-D interpolation procedure that maintains the continuity of the function and its first derivatives (both normal and tangential) across cell boundaries [102].

The bicubic algorithm applies convolution in the dimensions. For equally spaced data, the interpolation functions can be written in the form

$$g(q)=\sum_k c_k u\left(\frac{q-q_k}{h}\right) \tag{3-34}$$

where the c_k's are parameters with depend on the sampled data, and u is the interpolation kernels. The parameter of the kernel is chosen so that the interpolation function and the Taylor series expansion for the original function agree for as many terms as possible. For the cubic convolution interpolation, the solution for the interpolation kernel is

$$\begin{cases} (a+2)|q|^3-(a+3)|q|^2+1 & 0<|q|<1 \\ u(q)=a|q|^3-5a|q|^2+8a|q|-4a & 1<|q|<2 \\ 0 & 2<|q| \end{cases} \tag{3-35}$$

where a is usually set to -0.5 or -0.75. Figure 3-17 shows the results of bicubic interpolation.

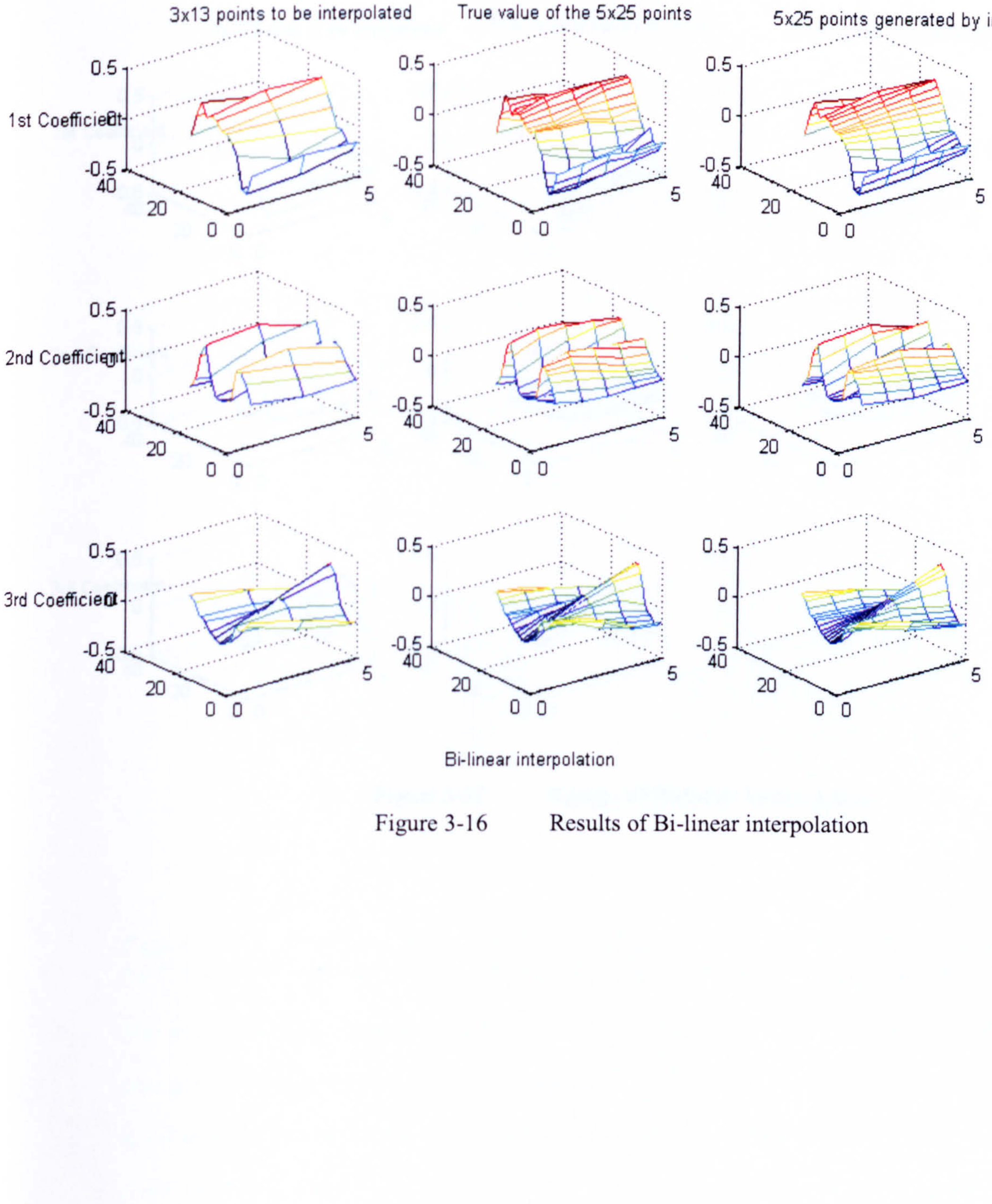


Figure 3-16 Results of Bi-linear interpolation

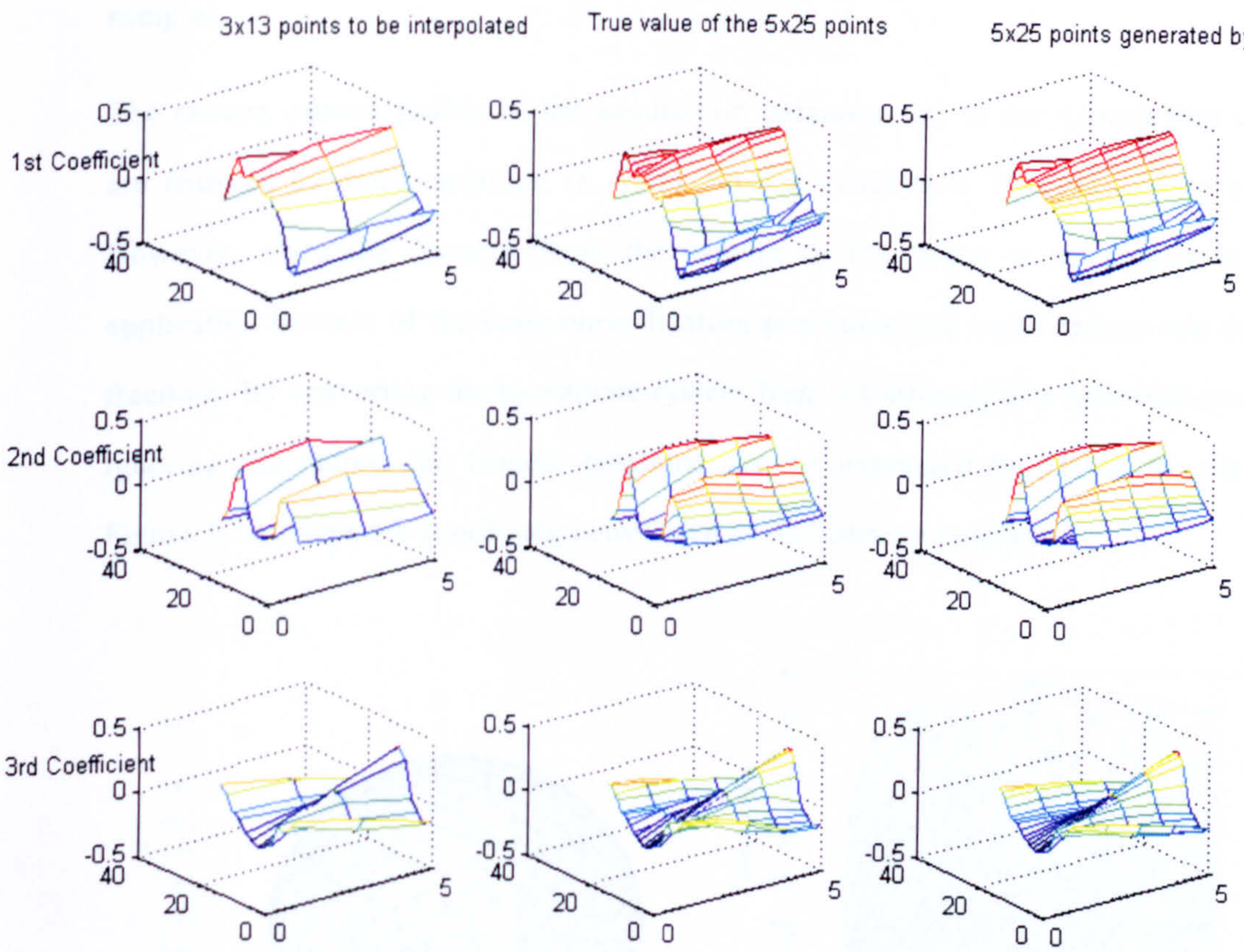
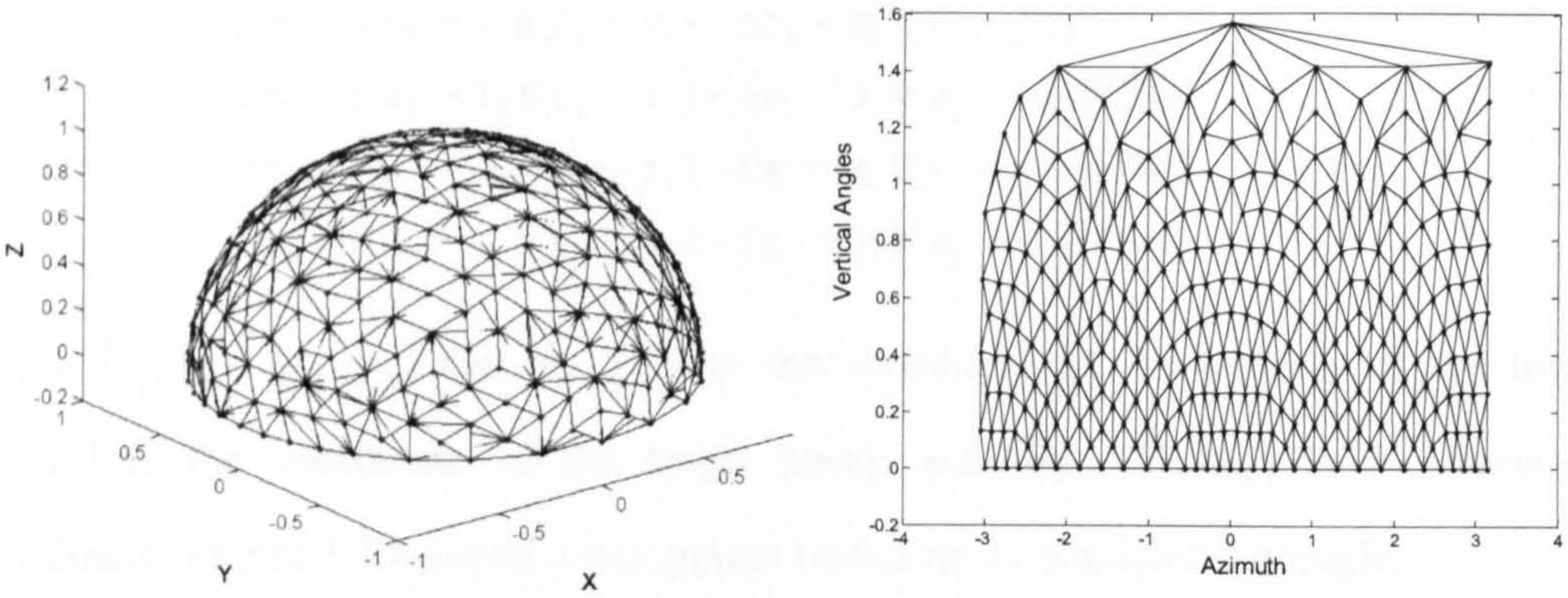


Figure 3-17 Results of Bi-Cubic Interpolation

In our Cameo-Sim database, the images are not samples on a uniform rectangular grid. Instead, the camera positions are on the vertices of a third-level Icosahedron (see Figure 3-38 (b)), upper sphere. To make the solution suit more general cases, we are not concentrating on any particular sampling structure, but on the most general one, scattered samples.

The chosen camera positions, the acquisition parameter q, in our Cameo-Sim database are featured by three variables, (x, y, z). So this could be a 3D interpolation problem. However, since the distance from the camera to the target is not important in our application because of the scale normalization procedure, we could reduce one degree of freedom. By converting the coordinate system from a Cartesian to a spherical system and ignoring the radius, the camera positions can be expressed by (azimuths, elevation). Figure 3-18 shows the coordinate conversion of 337 camera positions.



(a) (b)

Figure 3-18 Coordinate system conversion of upper sphere vertices in 3rd level Icosahedron. (a) Cartesian system (x,y,z); (b) Spherical system (azimuth, vertical angle), heights ignored

We use the matlab function 'griddata' to do the interpolation. This function can do both triangle based linear and cubic interpolation [103] [104] where the latter one can produce a smooth surface while the first one has discontinuities in the first derivative. Here we describe how triangle based interpolation works using the linear case as an example. The table below gives the main procedures in the linear interpolation:

Input: sample data coordinates (x, y), sample data value z, data coordinates after interpolation (xi, yi); **Output:** data value of points (xi, yi), zi;
1. Triangularize [105] all the sample data points (x, y); record all the triangles as *tri*
2. For each data point in (xi, yi), find the nearest triangle in *tri*
3. Keeps only the relevant triangles in *tri*
4. Compute the Barycentric coordinates wi, (wi_1, wi_2, wi_3), of each point in (xi, yi)
5. Compute zi by $zi_k = wi_{k1} z_{tri(k1)} + wi_{k2} z_{tri(k2)} + wi_{k3} z_{tri(k3)}$, where the $z_{tri(k1)}$, $z_{tri(k2)}$, $z_{tri(k3)}$ are the data value of the three vertices of the triangle associated with point (xi, yi)

Step 4 in the above table is to calculate the Barycentric coordinate for the points. As a reminder, the Barycentric coordinates [106] are coordinates defined by the vertices of a simplex. With respect to a triangle, the Barycentric coordinate can be calculated by:

$$\begin{aligned} Del &= (x_2 - x_1)(y_3 - y_1) - (x_3 - x_1)(y_2 - y_1) \\ wi_1 &= \big((x_2 - x_i)(y_3 - y_i) - (x_3 - x_i)(y_2 - y_i)\big)/Del \\ wi_2 &= \big((x_3 - x_i)(y_1 - y_i) - (x_1 - x_i)(y_3 - y_i)\big)/Del \\ wi_3 &= \big((x_1 - x_i)(y_2 - y_i) - (x_2 - x_i)(y_1 - y_i)\big)/Del \end{aligned} \tag{3-36}$$

where (x_1, y_1), (x_2, y_2) and (x_3, y_3) are the coordinate of the vertices of the triangle, (x_i, y_i) is the coordinate of the target point, and (wi_1, wi_2, wi_3) is the Barycentric coordinate. Figure 3-19 shows some typical points on an equilateral triangle.

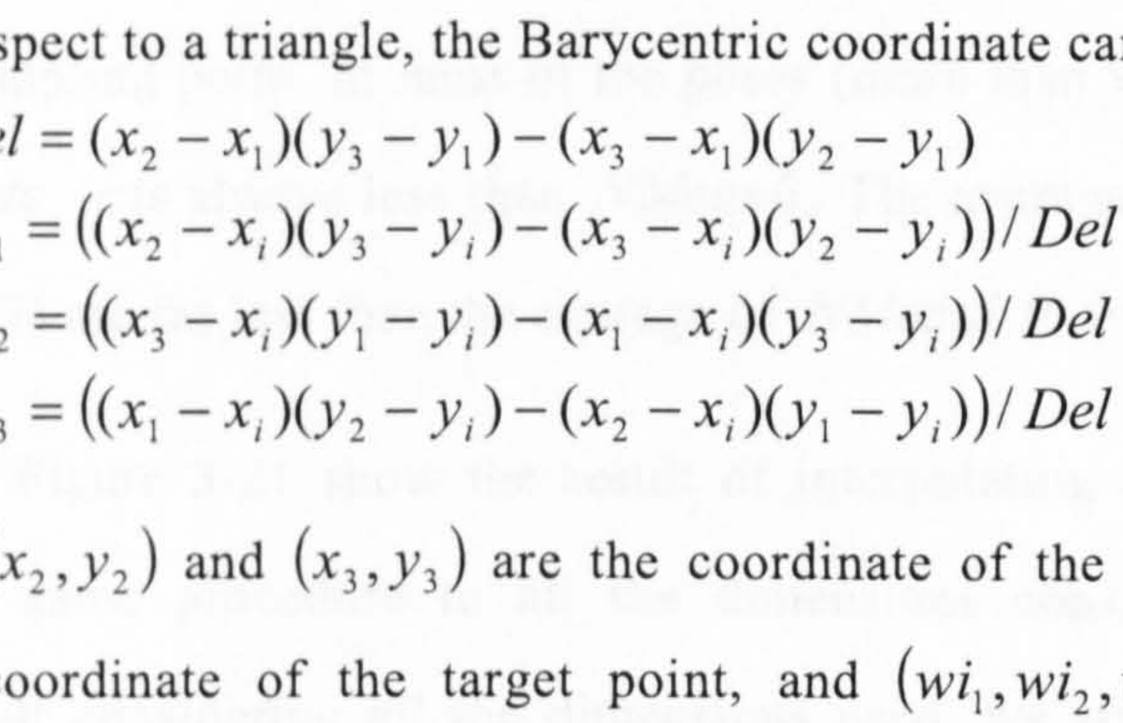
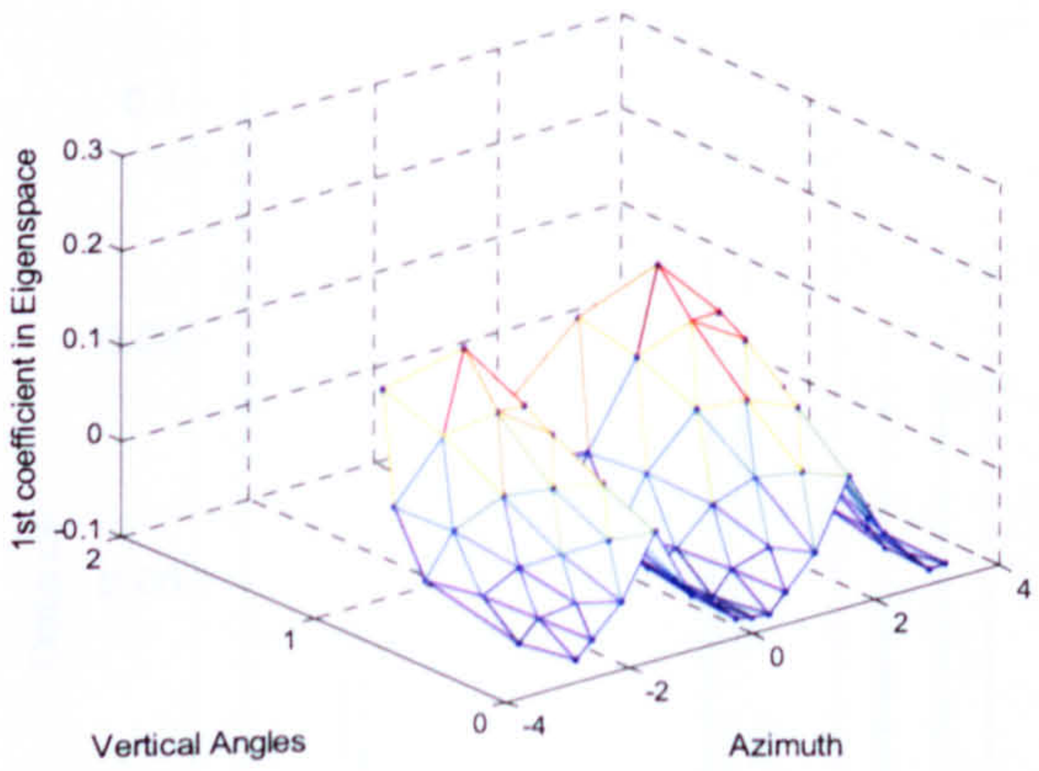


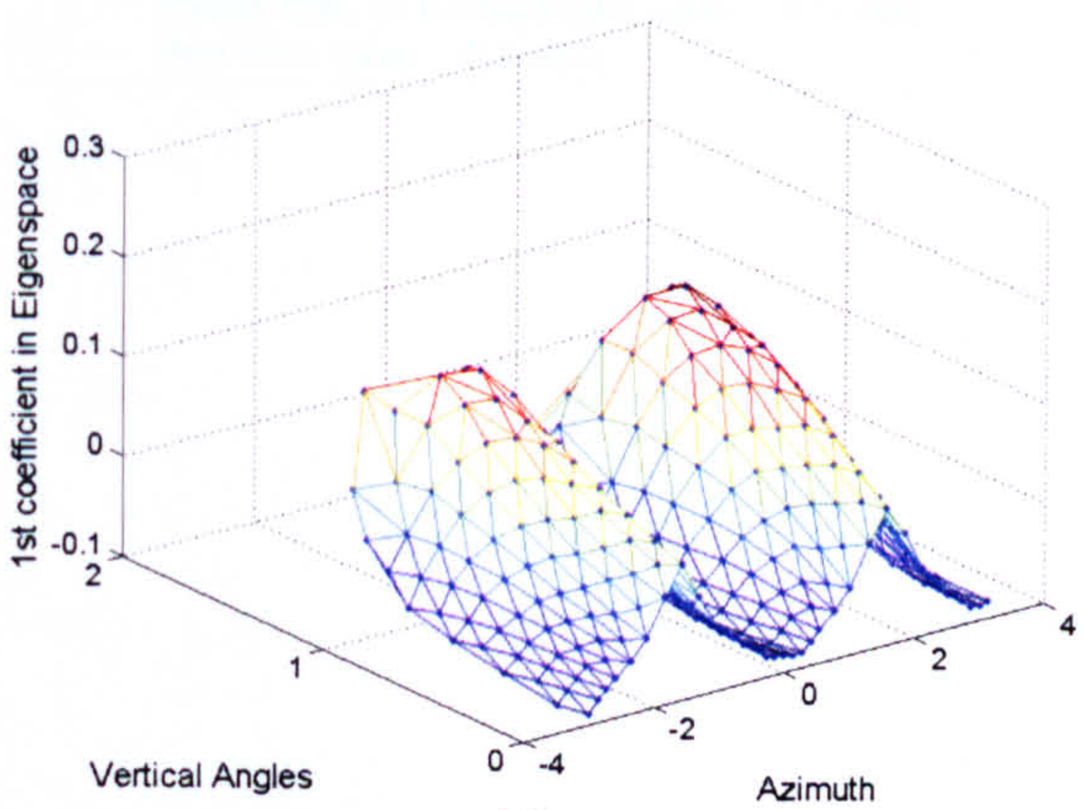
Figure 3-19 Barycentric coordinates on an equilateral triangle

To test the quality of the interpolation, we use 89 points, which are the upper sphere vertices of the 2nd level Icosahedron, as samples, and interpolate based on these 89 points to generate 337 points at the position of upper sphere vertices of the 3rd level Icosahedron. Figure 3-20 illustrates the results of linear and cubic interpolation, along with the sample data and the exact data. To evaluate the quality of the interpolation, we compare the interpolation error, *Inte_l*, and *Inte_c* (data generated by the interpolation minus the exact data) with the distance between the exact data and its nearest neighbour, *NMin* , and with the mean distance between the exact data and its 6 nearest neighbors, *Inte_l*. The *NMean*6 is chosen because in the view sphere, each points have 6 nearest neighbors with equal distances. If the interpolation error is less than *NMean*6, the interpolation method is generally acceptable. Figure 3-21 shows the results of the comparison. We see that the interpolation errors at the first 89 points are zero. This is because that those points are the original 89 sampling points and the resulting interpolation surface goes through every original sampling point. In most of the poses (more than 95%), *Inte_l* is less than *NMean*6, and *Inte_c* is always less than *NMean*6. The average of *Inte_l* (0.0039) and of *Inte_c* (0.0027) are far less than the average of *NMean*6 (0.0146).

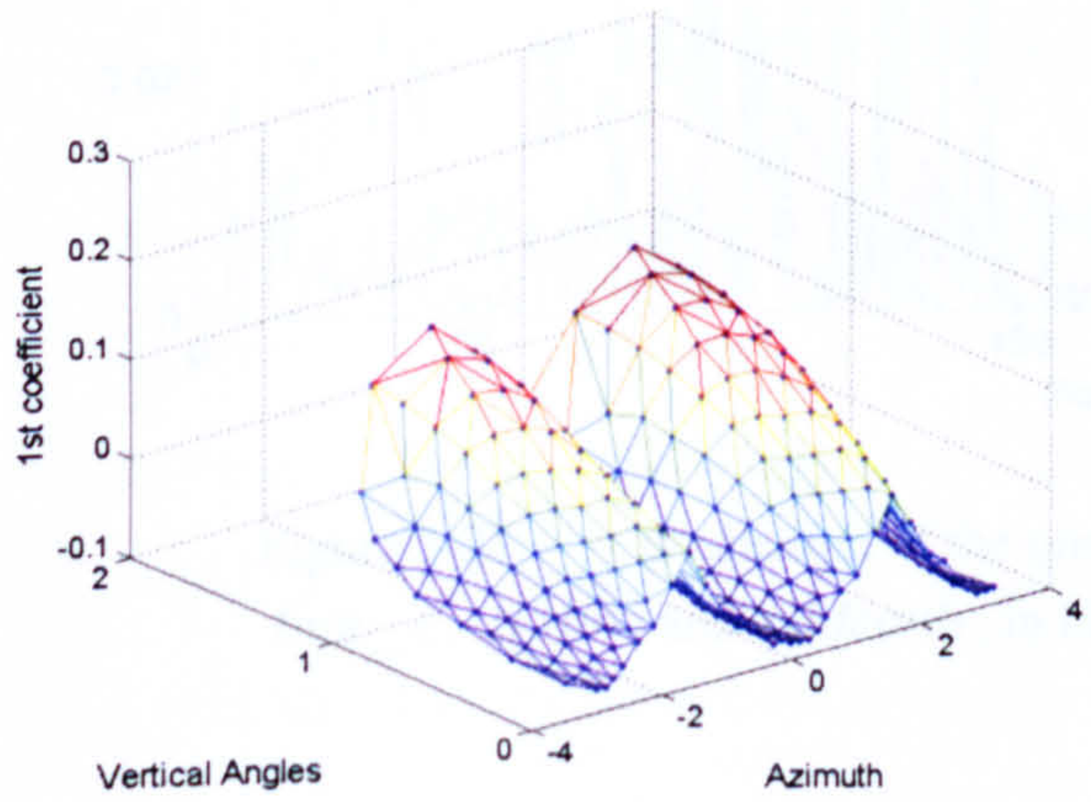
Figure 3-20 and Figure 3-21 show the result of interpolating the first coefficient. We could apply the same procedure to all the dimensions considered. To evaluate the interpolation result considering all the dimensions used, we still use *Inte_l*, *Inte_c*, *NMean*6, and *NMin*. The only difference is that we use Euclidean distance for higher dimensions. Figure 3-22 shows the comparison based on 100 dimensions and 11 objects. Each bar value is an average of (337-89) samples. We see that the cubic interpolation is very slightly better than the linear one. Both of the average *Inte_l* and *Inte_c* of all 11 objects are far below the *NMean*6 value, which means that both interpolation methods are acceptable.



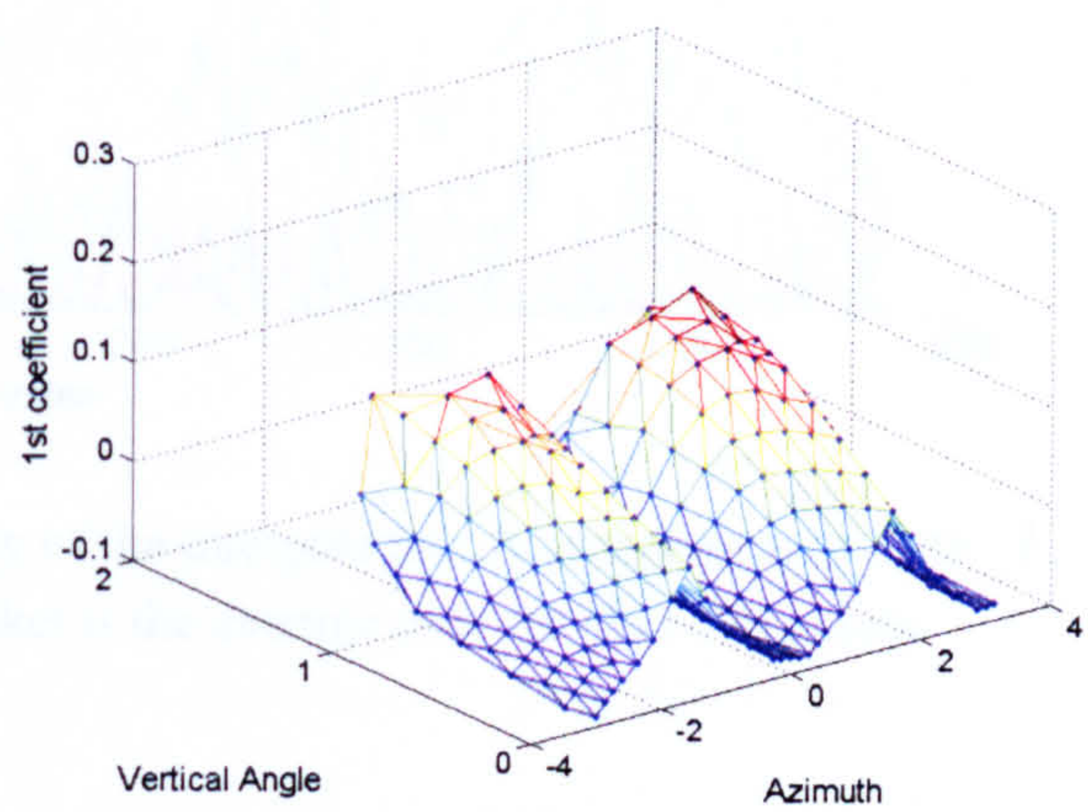
(a)



(c)



(b)



(d)

Figure 3-20 Interpolation of the 1st coefficient in Eigenspace. (data: 2nd object in Cameo-sim database, the Landrover) (a) Sampling points (89 points): upper sphere of 2nd level Icosahedron. (b) Exact value of the 337 points sampled from the upper sphere of 3rd level Icosahedron. (c) Result of cubic interpolation (d) result of linear interpolation

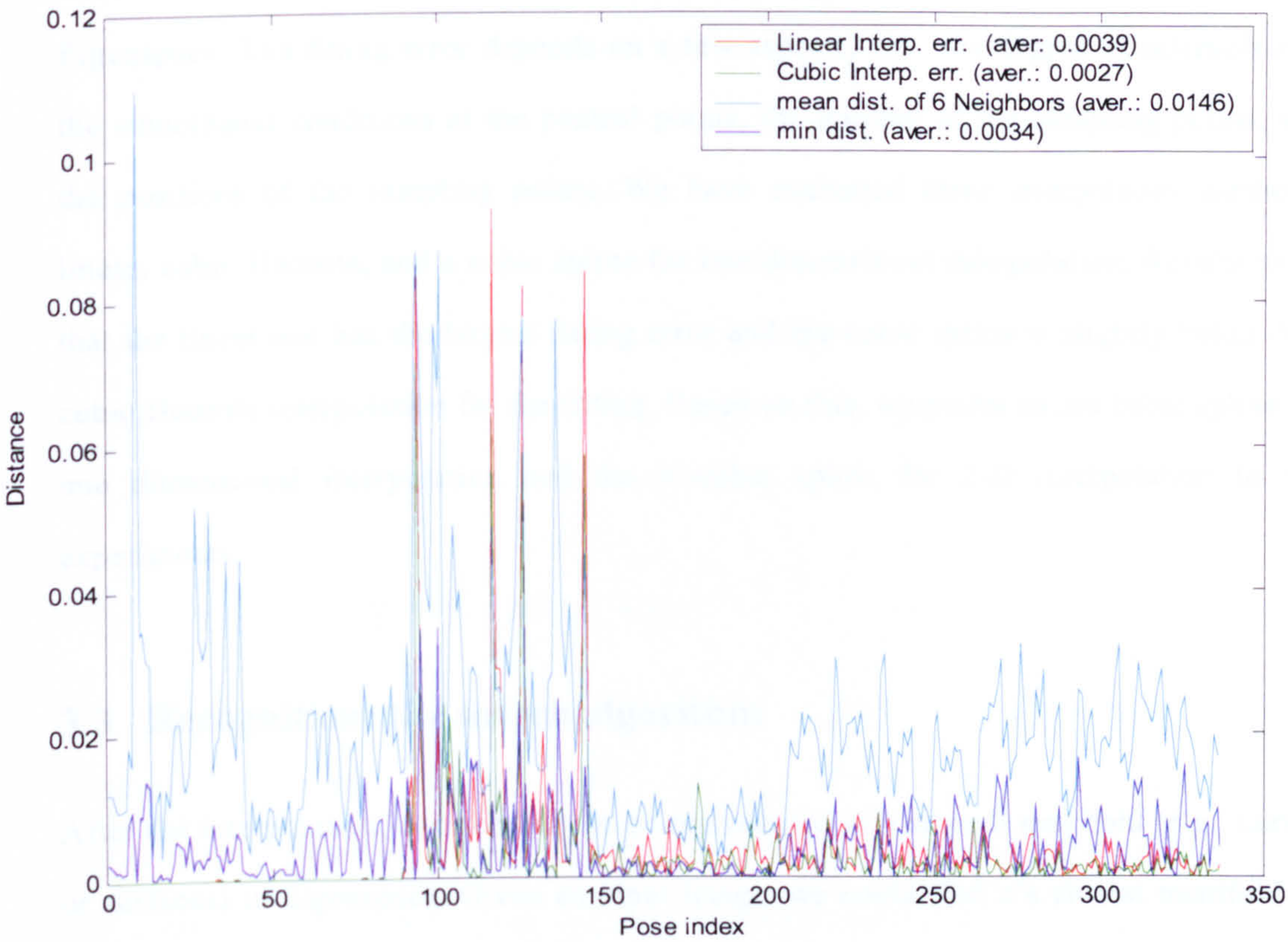


Figure 3-21 Evaluation of the quality of the interpolation: A comparison of *Inte_l*, *Inte_c*, *NMin*, and *NMean6*, in blanket is the average over the (337-89) points

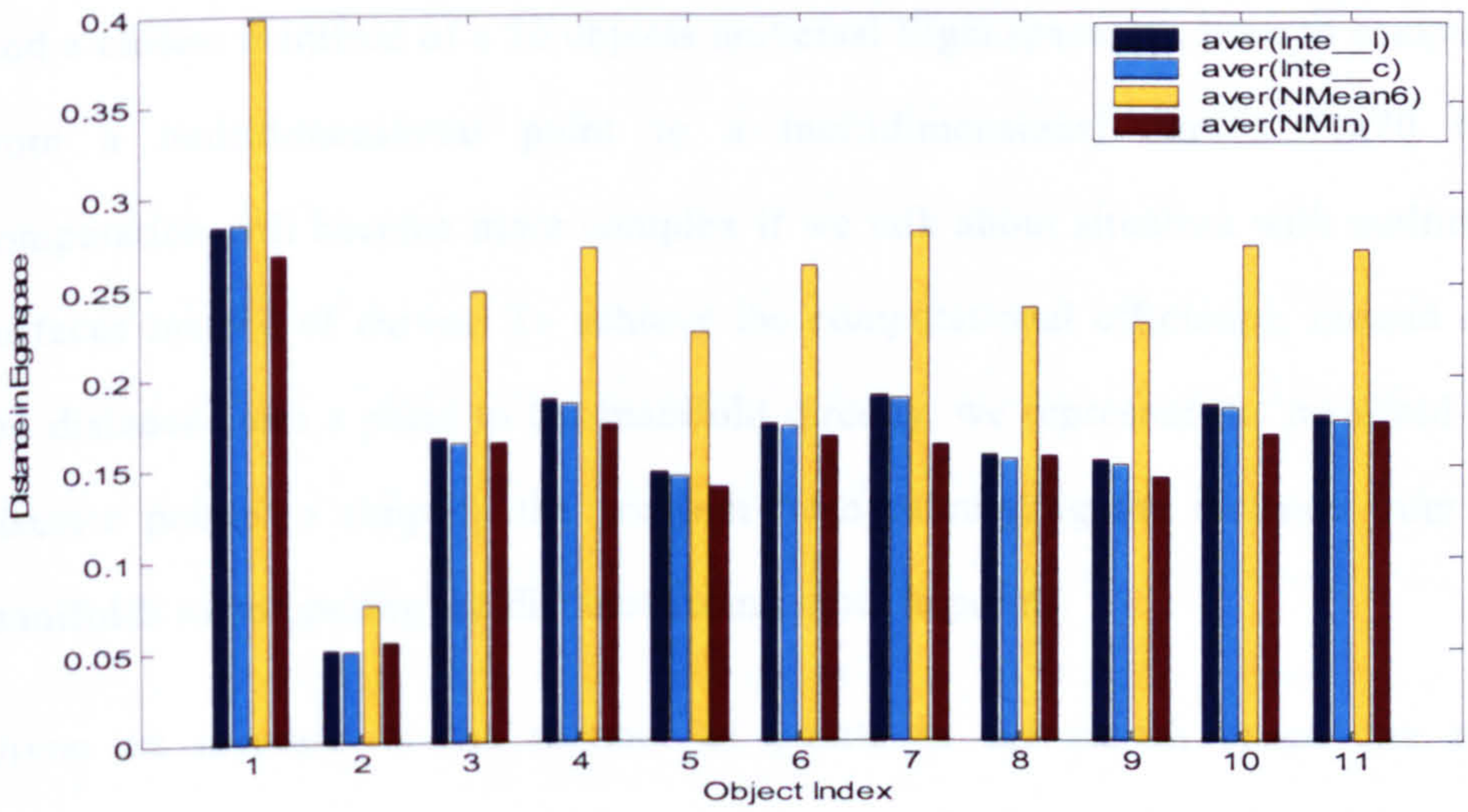


Figure 3-22 Evaluation of the quality of the interpolation for all dimensions. Each bar value is an average of (337-89) samples

In this section, we discussed several interpolation methods to fit the true coefficients in Eigenspace. The fitting error depends on a few aspects, e.g., the degree of interpolation, the smoothness conditions at the control points, the number of the sampling points, and the positions of the sampling points. We have evaluated three interpolation methods, linear, cubic Hermite, and a cubic spline for one dimensional interpolation. Results show that the linear one has the largest fitting error and the cubic spline is slightly better than cubic Hermite interpolation for the fitting. Based on this, we prefer to use cubic spline for one dimensional interpolation and the bi-cubic spline for 2-D interpolation in our experiments.

3.3 Recognition: the search algorithm

After the interpolation, each object can be represented as a smooth manifold (e.g., curves or surfaces) in Eigenspace. Given an input image, we could find it's closest manifold by calculate the distance between the projection of the input image and the manifold of known objects. However, each manifold is not defined by only one function, but a set of functions which connect smoothly at the joint points or edges. For example, an object in coil database is represented by 71 functions, each between two connected poses. Thus, to find a closest manifold of a 20 objects universal Eigenspace, we have to compute distance from a multidimensional point to a multidimensional curves 71x20 times. The computation will become more complex if we talk about situation with multidimensional surfaces instead of curves. To achieve the computational efficiency, instead of compute the distance from a point to the manifold directly, we represent the manifold by a set of discrete points to simplify the problem from calculating the distance from a point to manifolds to computing the distance from points to points.

Given an increase in the number of points in the search space due to B-spline interpolation, and possibly due also to an increase in the number of models and scenarios that must be considered, a new search strategy is desirable to reduce the complexity of the

matching procedure. Exhaustive search should always find the optimal solution, given the parameters of the problem, but to reduce our time and space complexity we need a better algorithm than exhaustive search. We use the idea of a multidimensional binary search trees posed by Bentley [107] (see section 3.3.1) to improve the efficiency. However, the searching result of this algorithm is not equivalent as the exhaustive search in section 3.1.6. This is because the way it measures the distance between two multidimensional points is not Euclidean distance. We propose a modification of the original multidimensional binary search trees method to solve this problem.

3.3.1 The basic algorithm and its complexity

We implement an alternative algorithm to perform an improved search through the multi-dimensional Eigenspace in $O(D \log_2 n)$, where n is the number of data points and D is the number of dimensions. This algorithm is based on the k-d tree structure, a natural generalization of the standard one-dimensional binary search tree.

A kd-tree (short for k-dimensional tree) is a space-partitioning data structure for organizing points in a k-dimensional space, in which every node from root to leaves stores a point. Figure 3-23 shows an example of the k-d tree structure in which $k = 3$. In Figure 3-23, node A is the root and all other nodes are leaves. There are totally 5 levels in the tree and each non-null node has two child leaves in the next level. In the right line of Figure 3-23, we see that each level can have different discriminators. In the special case of a k-d tree where k equals one, a binary tree, each node is represented by one number and that number is the key for constructing and searching the tree. However, in a k-d tree where $k \geq 2$, every node is represented by a vector. In fact, different elements in the vector provide the key in different levels of the tree. The discriminator determines which element, or dimension, is the key in any given level. Each node in a tree is associated with a discriminator. All nodes at any given level of the tree have the same discriminator. The root node has discriminator 0, its two child leaves have discriminator 1, and so on to the k th level at which the discriminator is $k-1$; the $(k+1)$ th level has discriminator 0, and

the cycle repeats. In our application, e.g., if we use a 10 dimensional Eigenspace, the number of levels of the tree varies depending on the number of training images and their relations; the discriminator starts from 0 at the root node and increases by 1 per level until it reaches 9 and the cycle repeats.

Our search algorithm consists of two parts: **constructing the tree structure** and **querying.** In the first part, we insert nodes one by one into an initially empty tree. The pseudo code of inserting a node to a tree is as follows:

```
Current_node = Root;
Discriminator = 0;
Recursive_insert (Current_node, Discriminator, Point)
{       if  Current_node = null;
                Current_node = Point;
        else
        {       if  Cuttent_node . data[Discriminator] < Point . data[Discriminator]
                        Recursive_insert(Current_node . right, (Discriminator +1)%K,
Point)
                else
                        Recursive_insert(Current_node . left, (Discriminator +1)%K,
Point)
        }
}
```

Here we give an illustration of how the construction procedure works. In the training stage, given A, B, C, D, E, F, G are seven 3D Eigenspace points, each of which is a projection of a training image into Eigenspace, Figure 3-23 shows how these 7 points are inserted as a sequence into an initially empty tree. Point A is inserted as the root of the tree. When we insert point B, we compare the value of the first dimension of B. Since it is larger than the first dimension of A, B is inserted as the right child leaf of A. When we insert C, as the first element of C is larger than that of A; the second element of C is then compared with the second element of B. C is finally inserted to the left child leaf of B. Points D, E, F and G are then inserted in the same manner. We see that the final structure of the tree may be different for a given set of points depends on the order we insert them.

In the second part, querying, we use a range search algorithm: for every new point to be compared with the node in the tree, we set a lower boundary and upper boundary, *lowk* and *highk*. Both of them are of the same dimensionality as the new point and nodes in the tree. In *lowk*, the values are lower then the new point in each dimension, while in *highk*, the values are higher in each dimension. The values of *lowk* and *highk* are set depending on the application. In our case, if we use categorical identification (see section 3.1.6), *lowk* and *highk* depend on the threshold we set. In comparative identification, we set a small range between *lowk* and *highk* first and increase the range gradually until we find the nearest neighbour. The following pseudo code represents a basic algorithm that searches a k-d tree for values contained between *lowk* and *highk*.

```
Cu_node = Root;
Discr = 0;
Re_Search (lowk, highk, Cu_node, Discr)
{       if  lowk[Discr] <= Cu_node . data[Discr]
                Re_Search (lowk, highk, Cu_node . left, (Discr+1)%K)
        for (j=0; j<K && lowk[j] <= Cu_node . data[j]
                        && highk[j] >= Cu_node . data[j]; j++)
        if (j = = K)
                FoundIt( );
        if  highk[Discr] > Cu_node . data[Discr]
                Re_Search (lowk, highk, Cu_node . right, (Discr+1)%K)
}
```

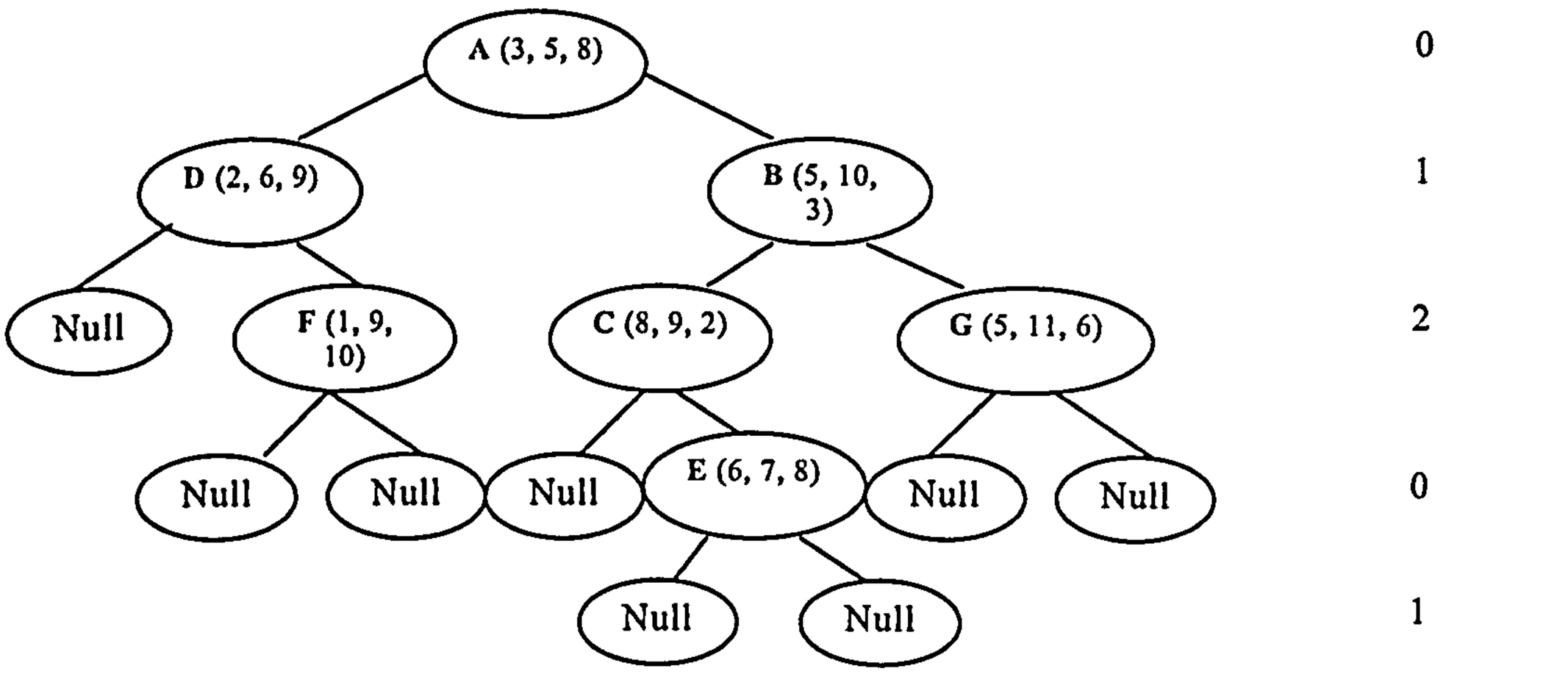


Figure 3-23 An example of records inserted as nodes in a 3-d tree. Points A, B, C, D, E, F, G are inserted as a sequence into an initially empty tree.

Here we also give an example of the querying. Consider the k-d tree in Figure 3-23, if the new point is [2 8 9] and have *lowk* and *highk* of [1 7 8] and [3 9 10] respectively. First we compare the first element of *lowk* with the root. Since it is lower than that of the root, we then compare the second element of *lowk* with node D. Since is not lower, we then check if node D is in the range of *lowk* and *highk*. The result is not. We then compare the second element of *highk* with node D – higher –> compare the third element of *lowk* with F – not lower –> check if F is in the range of *lowk* and *highk* – yes –> find that the node F is the nearest node to the new point.

Bentley [107] proved that the probability of constructing a k-d tree by inserting n random nodes into an initially empty k-d tree is the same as the probability of attaining that tree by random insertion into a one-dimensional binary search tree. Let C_n be the number of nodes visited to find a node in a k-d tree with n nodes. Then the mean of the distribution of C_n is:

$$Mean(C_n) = 2(1+1/n)H_n - 3 \approx 1.3863\log_2 n \tag{3-37}$$

and the variance is:

$$Var(C_n) = 7n^2 - 4\,(n+1)^2 H_n^{(2)} - 2(n+1)H_n + 13n \tag{3-38}$$

(a) (b)

Figure 3-24 An exhaustive search within a hypercube may yield an incorrect result. (a) P_2 is closer than P_1, but a search based solely on the hypercube will incorrectly identify P_1 as the closer point. (b) This can be remedied by forming another hypercube which bounds the hypersphere. The closest existing point inside this hypercube must be the closest in the whole space.

where $H_n^{(x)} = \sum_{i=1}^{n} 1/i^x$ and $H_n = H_n^{(1)}$. Thus we know that a typical insertion or retrieval in a k-d tree will examine approximately $1.386 \log_2 n$ nodes.

3.3.2 Problem of the basic algorithm and a proposed solution

Searching a k-d tree is analogous to searching a hypercube in a high dimensional space. However, as seen in Figure 3-24, this does not always correctly find the closest point, especially in higher dimensions. Points outside the hypercube can be closer than points inside. The term closest point refers to the point with the minimum distance from the control point. We need to define the distance metric between two points.

The L_p distance between two n-dimensional vectors a and b is defined as

$$L_p(a,b) = \left[\sum_{k=1}^{n} (a_k - b_k)^p \right]^{1/p} \tag{3-39}$$

Where a_k and b_k are the kth dimension of vector a and b and p is the Minkowski factor for the norm. As illustrated in Figure 3-25, these distance metrics are also known as Minkowski metrics [109]. Particularly, when p is set as 2, it is the well known Euclidean distance; when p is 1, it is the Manhattan distance (or L1 distance).

The Euclidean distance occurs most frequently in pattern recognition problems. From the last section, we see that the distance used between points in the multidimensional space is

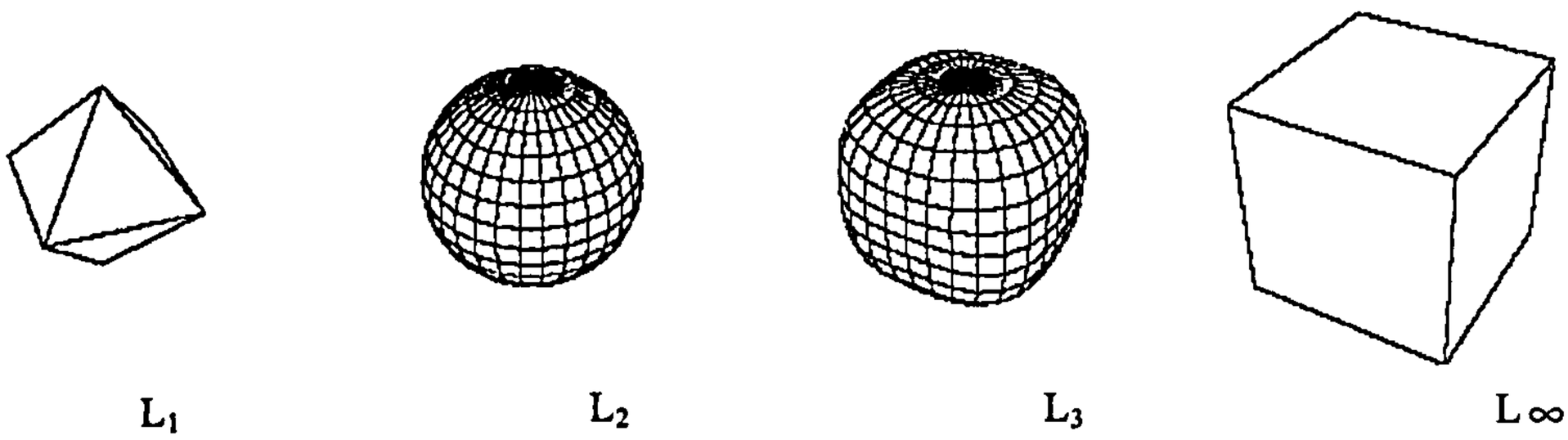


Figure 3-25 An illustration of various norms, also known as Minkowski p-metrics. All points on these surfaces are equidistant from the central point.

the Euclidean distance. Unfortunately, the hypercube approach is to find the points within L_∞.

As we increase dimensionality, the difference between the hypersphere and hypercube becomes so great that the hypercube "corners" contain far more points than the inscribed hypersphere. Let d be the dimensionality and r be the radius of the hypersphere and half-length of one side of the hypercube. The volume of a hypercube is

$$V_c = 2^d r^d \tag{3-40}$$

The volume of a hypersphere is

$$d = 2k, \qquad V_s = \frac{\pi^k r^{2k}}{k!}$$

$$d = 2k+1, \quad Vc = \frac{2^{2k+1}\pi^k r^{2k+1} K!}{(2k+1)!} \tag{3-41}$$

Hence, the ratio of the volume of a hypersphere to a hypercube with the same d and r is given by

$$d = 2k, \qquad P = \frac{\pi^k}{k!2^k}$$

$$d = 2k+1, \quad P = \frac{k!\pi^k}{(2k+1)!} \tag{3-42}$$

This is plotted in Figure 3-26. For d = 2, the ratio (proportion) is 0.7854. For d = 20, the ratio is 2.4611×10^{-8}. This means that in a high dimensional search space, the probability of successfully finding the nearest point by using a simple kd-tree search based on a

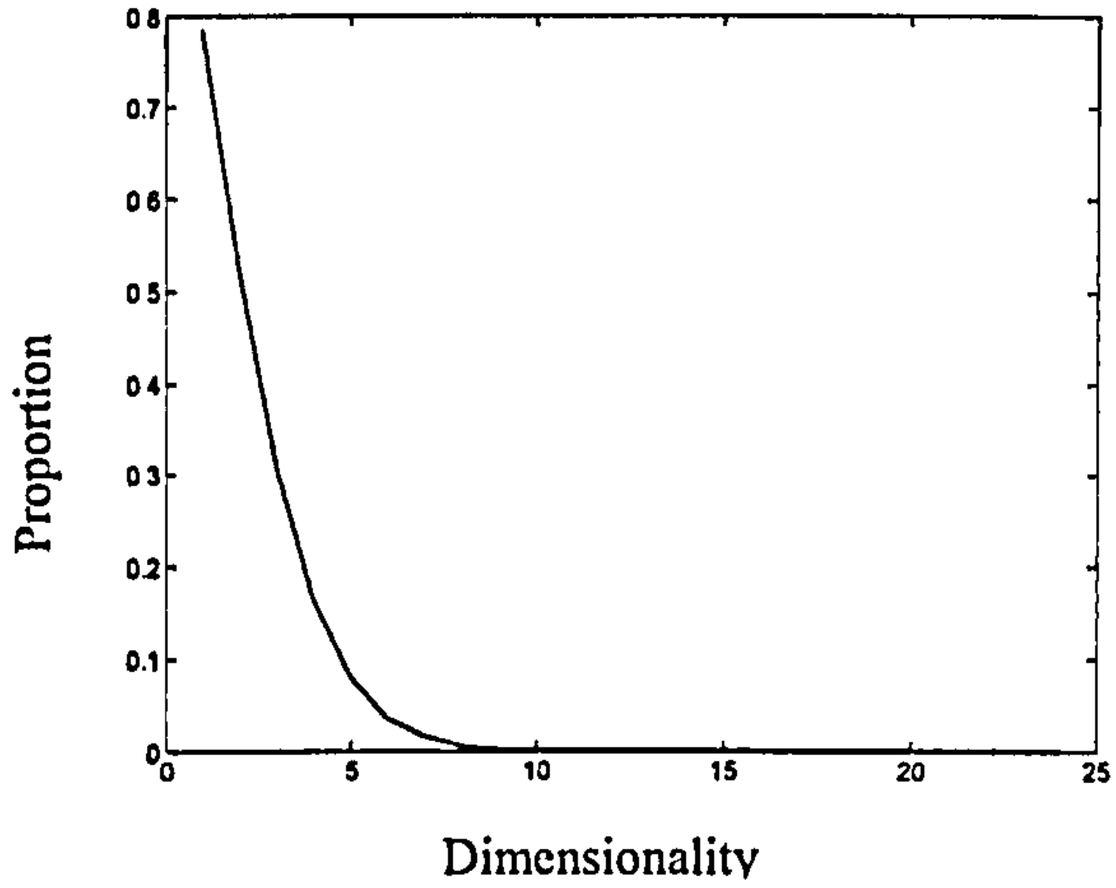


Figure 3-26 As the dimensionality increases, the ratio of the volume of a hypersphere to the bounding hypercube decreases dramatically.

hypercube is low. For example, Murase and Nayar [69] determined that a universal Eigenspace with 10 dimensions was sufficient to get near perfect recognition performance (for their particular, limited data set). For d=10, if the points are uniformly distributed in the 10 dimensional space, the probability of finding a point at the "corners" of the hypercube rather than in the central hypersphere is 93.04%.

Since it is not sufficient to simply search for the closest point within a hypercube because a point outside can be closer than a point inside, we suggest the following technique. First, a k-d tree range search is performed to compute the points within the hypercube C_1 with a half side length of r, as illustrated in Figure 3-24(b). The closest point we find is point P_1, which is located at the corner of C_1. Clearly, if a closer point exists, it can only be within a hypersphere S of radius $\sqrt{k}r$, where k is the dimensionality. Since part of S lies outside the original hypercube C_1, we construct another hypercube C_2 which is the minimum hypercube that contains S and do a k-d tree range search for the points which lie in the hypercube C_2. As shown in Figure 3-24(b), three points, $P_1, P_2, andP_3$, are found. Finally, we perform an exhaustive search within these 3 points to find the nearest one, P_2. The modified algorithm is shown as follows:

A Modified Search Algorithm

1. K-d tree range search in a range of *lowk* [O_1-r, O_2-r, ..., O_k-r] and *highk* [O_1+r, O_2+r, ..., O_k+r]. Adjust r to find more than one point in the range.
2. Increase r to $r_{new} = \sqrt{k}r$.
3. K-d tree range search in a range of $lowk_{new}$ [O_1-r_{new}, O_2-r_{new}, ..., O_k-r_{new}] and $highk_{new}$ [O_1+r_{new}, O_2+r_{new}, ..., O_k+r_{new}].
4. Exhaustive search among points found in step 3 to find the closest point.

3.4 Robust Sampling Method

3.4.1 Problems in standard appearance-based recognition

The major advantage of ordinary appearance-based recognition is that both learning and recognition are performed using just brightness images without any low- or mid-level processing. However, several problems arise because the technique relies on the direct use of a large set of images, usually of different poses of the object, for which the intensity values may vary considerably. The most severe limitations of the method in its standard form are that it cannot handle problems related to occlusion and varying background:

1. Occlusion: Suppose that the image is occluded by a black band: $x = [x_1,...x_r,x_{r+1},...x_m]^T$ becomes $\hat{x} = [x_1,...x_r,0,...,0]^T$,

then $$\hat{a}_i = \hat{x}^T e_i = \sum_{j=1}^{r} x_j e_{i,j}.$$

The error we make in calculating a_i is

$$(a_i(x) - \hat{a}_i(\hat{x})) = \sum_{j=r+1}^{m} x_j e_{i,j}.$$

It follows that the reconstruction error is

$$\left\| \sum_{i=1}^{p} \left(\sum_{j=r+1}^{m} x_j e_{i,j} \right) e_i \right\|^2.$$

Figure 3-27 illustrates the effect of occlusion on the reconstructed image.

2. Varying Background: In calculating the eigenimages no distinction is made

Original image　　Occluded image　　Reconstructed image

Figure 3-27　Demonstration of the effect of occlusion using the standard approach for calculating the coefficients.

between the object and the background. Therefore, the effect of a varying background is similar to that of occlusion.

Different approaches have been proposed in the literature to estimate the coefficients of the Eigenspace projections more reliably. Ohba and Ikeuchi [111] proposed the eigenwindow method to recognize partially occluded objects. The methods based on "eigenwindows" do not solve the problems entirely because the same limitations hold for each of the eigenwindows. Besides, due to local windows, these methods lack the global aspect and usually require further processing.

To eliminate the effects of varying background, Murase and Nayar [112] introduced the search-window, which is the AND area of the object regions of all images in the training image set. However, the assumption on which the method has been developed is rather restrictive; namely, a target object can only be occluded by one or more of the other target objects, rather than occluded by some unknown entity or perturbed by a different background.

3.4.2 Theory of Random Sampling

Previous experiments have shown that the appearance based method is not good at recognizing the occluded object in a test image. An intuitive approach to removing the effects of occlusion might be summed up by the following question — can we use only the non-occluded pixels of the image instead of using the whole image?

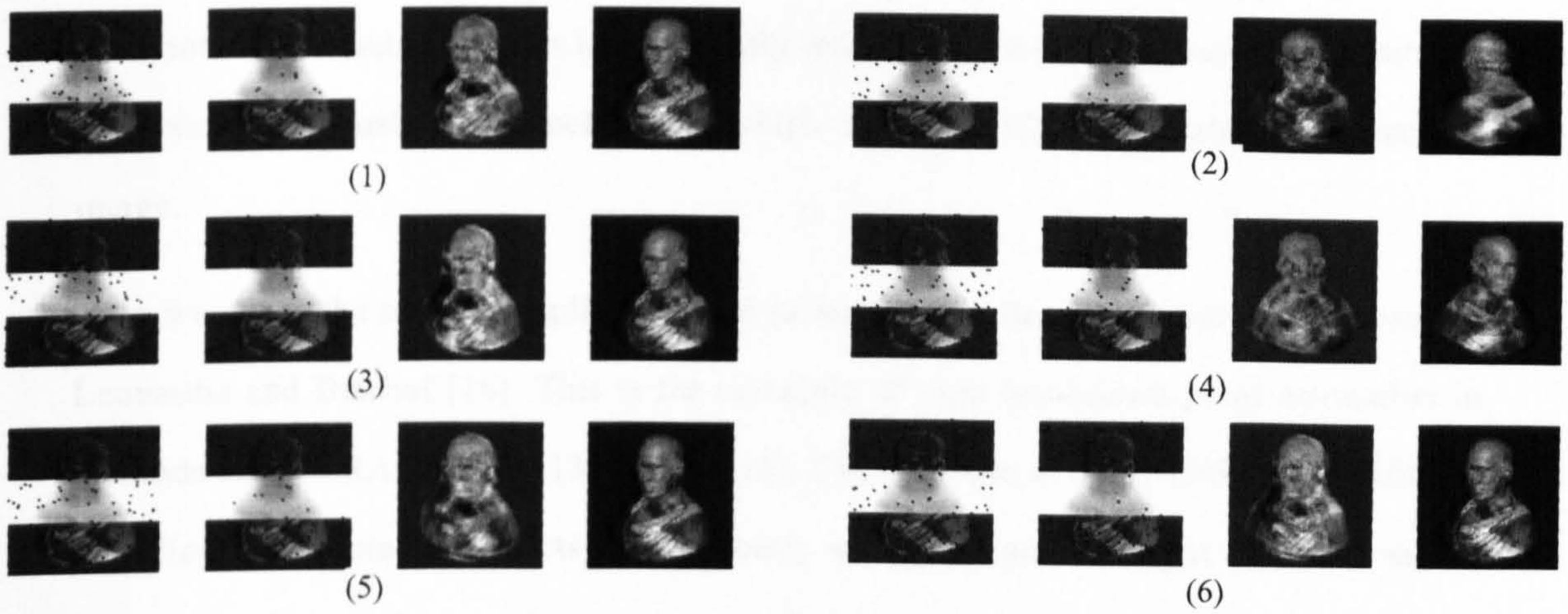


Figure 3-28 Some hypotheses generated by the robust method for the occluded image with 9 eigenimages; for each hypothesis (1–6), from left to right: reconstructed image based on the initial set of points, reconstruction after reduction of 25% of points with the largest residual error, and the reconstructed image based on the parameters of the closest point on the parametric manifold.

First, we need to consider whether a subset of pixels is sufficient for recognition in the appearance based approach. Given a noisy or occluded image to be recognized, is it possible to ignore or remove the damaged part? Let's start with the representation of the image by a combination of eigenimages:

$$\vec{x} = c_1\vec{e}_1 + c_2\vec{e}_2 + \cdots + c_n\vec{e}_n \tag{3-43}$$

Here, $\vec{x}$ is the original image with m pixels to be recovered, c_i are the coefficients, e_i the eigenimages, and n is the total number of eigenimages ($m >> n$). To simplify the notation we assume the $\vec{x}$s to be normalized, having zero mean. Equation (3-43) can be treated as m independent equations. In the recognition stage, the $\vec{x}$ and e_i are known, and the first task is to determine the coefficients c_i. However, in order to calculate these, we need only n equations. Herein lies the possibility of an answer to the question at the beginning of this paragraph. The coefficients of the input image can be estimated from a small subset of pixels by solving an over-determined system of linear equations.

Instead of computing the coefficients by a projection of the whole image data onto the eigenimages (which is equivalent to determining the coefficients in a least-squares manner), the robustness can be achieved by subsampling. Therefore, if the selected subset

does not contain outliers and it is sufficiently representative, the obtained coefficients are approximately equal to the coefficients which would be obtained from the outlier-free image.

Here we adopt the robust sampling method in appearance based recognition proposed by Leonardis and Bischof [16]. This is the principle of high breakdown point estimation in methods such as RANSAC[5] [113] [114] [115]. The structure of the RANSAC algorithm is as follows. Repeatedly, subsets are randomly selected from the input data and model parameters fitting the sample are computed. The size of the random samples is the smallest sufficient for determining model parameters. In a second step, the quality of the model parameters is evaluated on the full data set. Different cost functions may be used for the evaluation. The standard one is the number of inliers, i.e., the number of data points consistent with the model.

Starting from the randomly selected k points $r_1,\ldots, r_k$, we seek the solution vector which minimizes Equation (3-44) in a least-squares manner. Then, based on the error distribution of the set of points, we keep reducing their number by a factor, α, (i.e. those points with the largest error) and solve Equation (3-44) again with this reduced set of points.

$$E(r) = \sum_{i=1}^{k} (x_{ri} - \sum_{j=1}^{p} a_j(x) e_{j,ri})^2 \qquad (3\text{-}44)$$

Figure 3-28 shows the progress of this algorithm. First, a subset of pixels was randomly chosen and the coefficients were calculated by solving an over-determined system of linear equations. Then, based on the error distribution of the subset of pixels, the number of selected pixels was reduced by a predefined factor f_1, the points with the largest reconstruction error being excluded and Equation (3-43) solved again with the reduced set of pixels. This procedure was repeated until the number of pixels reaches a predefined number n_1. After that, the whole image was reconstructed, the pixels with an error less than f_2 added to the subset, and the coefficients recalculated until the number of the pixels in the subset was stable.

---

[5] RANdom SAmple Consensus

However, the robust procedure does not guarantee the production of a good hypothesis from the initial, randomly chosen set of pixels. In the selection step, the best hypothesis is chosen, i.e., that hypothesis with the smallest reconstruction error in the compatible points.

The set of hypotheses, described by the coefficient vectors $\hat{c}$, the $\hat{e}$ vectors error, and the domains of the compatible points, which have been generated are often largely redundant. In order to select a subset of "good" hypotheses and reject the superfluous ones, a method described in the literature [116] [117] which leads to the minimization of an objective function encompassing the information on the competing hypotheses may be employed.

The objective function has the following form:

$$F(h) = h^T Ch = h^T \begin{bmatrix} c_{11} \cdots c_{1R} \\ \vdots \quad\quad \vdots \\ c_{R1} \cdots c_{RR} \end{bmatrix} h \tag{3-45}$$

Vector $h^T = [h_1, h_2, \cdots, h_R]$ denotes a set of hypotheses, where h_i is a presence-variable having the value 1 for the presence and 0 for the absence of the hypothesis i in the resulting description. The diagonal terms of the matrix C express the cost–benefit value for a particular hypothesis i

$$c_{ii} = K_1 s_i - K_2 \|\xi_i\| - K_3 N_i \tag{3-46}$$

where s_i is the number of compatible points, $\|\xi_i\|$ is the error in the hypotheses and N_i is the number of coefficients (eigenvectors). The off-diagonal terms handle the interaction between the overlapping hypotheses.

To simplify the situation, we only consider the diagonal term, in which K_1 is the average number of bits which are needed to encode an image when it is not encoded by the Eigenspace, K_2 is related to the average number of bits needed to encode a residual value of the Eigenspace approximation, and K_3 is the average cost of encoding a coefficient of the Eigenspace. Due to the nature of the problem, i.e. finding the maximum of the objective function, only the relative ratios between the coefficients play a role, e.g.,

$K_2 / K_1, K_3 / K_1$ and, moreover, $K_3 N_i$ is same for all hypothesis. Thus, c can be reduced to c':

$$c_{ii}' = s_1 - (K_2 / K_1)\|\xi_i\| \tag{3-47}$$

3.4.3 Implementation of the robust algorithm

Recognition Stage[6]

Input: test image x_{inl}, eigenvectors for each individual Eigenspace[7] E_i, mean image vectors for each object M_i, coefficient vectors of the training samples C_{jl}

Output: coefficient vector of the test image c, object ID d

Generating Hypotheses:

1: **for** *each individual Eigenspace*
2: **repeat**
3: Randomly choose n pixels $\hat{x}$ from x_{inl} - M_i
4: **repeat**
5: Calculate the coefficient vector from the subset of pixels[8]
6: Reconstruct the selected pixels $\hat{r} = E' \bullet \hat{c}$
7: Calculate the reconstruction error $err = |\hat{r} - \hat{c}|$
8: Retain the pixels with the smallest reconstruction error using a factor f_1<1: $n = n * f_1$
9: **until** n reaches a predefined number n_1
10: **repeat**
11: Reconstruct the whole image $\hat{i} = E \bullet \hat{c}$
12: Choose the pixels with the smallest reconstruction error less than f_2 and add them to $\hat{x}$
13: Recalculate the coefficient vector $\hat{c} = E'^{T}_{i} \bullet \hat{x}$
14: **until** the change in $\hat{x}$ is small
15: **until** the number of hypothesis reaches f_3
16: **end**

Selecting Hypothesis (a simple one)

17: **for** each hypothesis
18: Calculate the reconstruction error (as stated in step 7) corresponding to the specific Eigenspace
19: Calculate $d = n - K_2 \sum err$
20: **end**
21: Choose the hypothesis corresponding to largest d

---

6 The robust sampling method differs only from the standard appearance method in the recognition stage.
7 In this robust approach, we build an individual Eigenspace for each object in the training stage
8 E_i is reduced, for each dimension of eigenvectors, retaining the same position as the chosen pixel's position in the text image.

3.5 Probabilistic Eigenspace

Moghaddam and Pentland [14] have derived a quantitative measure of how well a test image fits an Eigenspace in terms of an estimated likelihood. They start by assuming that the training images have a high-dimensional Gaussian distribution, treating each image as a vector. The dimensionality of the Gaussian distribution is the number of pixels in each image. Under this assumption, the likelihood that an input pattern belongs to a trained class Ω can be written as the standard multivariate Gaussian density:

$$P(x \mid \Omega) = \frac{\exp\left[-0.5(x-\bar{x})^T \Sigma^{-1}(x-\bar{x})\right]}{(2\pi)^{N/2}\,|\Sigma|^{1/2}} \tag{3-48}$$

where $\bar{x}$ is the mean and Σ is the covariance of the training set $\{x^t\}$. Since everything else is constant, the sufficient statistic for characterizing this likelihood is the Mahalanobis distance:

$$d(x) = \tilde{x}^T \Sigma^{-1} \tilde{x} \tag{3-49}$$

where $\tilde{x} = x - \bar{x}$. This distance determines the similarity of a test image to the training image set. It differs from Euclidean distance in that it takes into account the correlations of the data set. Using the Eigenvectors and Eigenvalues of Σ, the Σ^{-1} can be rewritten in a diagonalized form and the Mahalanobis distance can be written as

$$d(x) = \sum_{i=1}^{N} \frac{y_i^2}{\lambda_i} \tag{3-50}$$

where y_i is the i th coefficients in Eigenspace, and λ_i is the corresponding Eigenvalue. Moghaddam and Pentland then divide the summation into two independent parts corresponding to the principal subspace and its orthogonal complement:

$$d(x) = \sum_{i=1}^{M} \frac{y_i^2}{\lambda_i} + \sum_{i=M+1}^{N} \frac{y_i^2}{\lambda_i} \tag{3-51}$$

The first term in Equation (3-51), referred to as Distance In Feature Space (DIFS), can be calculated directly by projecting the input image to the Eigenspace formed by the training images; the second term, referred to as Distance From Feature Space (DFFS), is rarely

computed explicitly in practice because of the high computational cost to deal with the high-dimensionality. To reduce this complexity, they estimated the DFFS using the reconstruction error between the true unknown image and the image reconstructed by the first M Eigenvectors, divided by a factor.

The assumption of Gaussian distribution is right in cases where the training images are accurately aligned views of similar objects seen from a standard view, e.g., frontal views of human faces. For cases where the training set represents multiple views or multiple objects under varying illumination conditions, Moghaddam and Pentland generalized the distribution of the first term in Equation (3-51) to arbitrarily complex distributions using a Mixture-of-Gaussians density model. They tested this Mixture-of-Gaussian model for face recognition using training images from different persons, views and illumination conditions and for hands detection using training images of different gestures.

In our application, the recognition of 3D objects, the training images of an object are taken from different views, each view is represented by one image only. This is not like the case of face detection and recognition where each view of a person's face is represented by many training images which could be assumed to have a Gaussian distribution. It is also not correct to use the Mixture-of-Gaussians model because sometimes the number of samples (training images) is small, e.g., in the Coil database, each object is represented by 72 views and unlike face recognition, different objects don't normally share similar views.

Therefore, in this section, we propose a probabilistic framework for general 3D object recognition. We adopt Moghaddam and Pentland's framework of dividing the probability into two parts, DIFS and DFFS, and because the latter one is not in a distance form, we call them the In-Space Error and the Out-of-Space Error. For the first part, we propose a simple method to calculate probability which is suitable for our training image set as described in the last paragraph. The Probabilistic Eigenspace Model described in this chapter can be used within an object Eigenspace because it gives quantitative results of how closely an unknown image is related to each object.

3.5.1 Calculating Image Likelihood

Given a test image I and an object model M, we would like to know whether the image contains the object being modelled. The probability of a model M given image I can be calculated using Bayes Theorem:

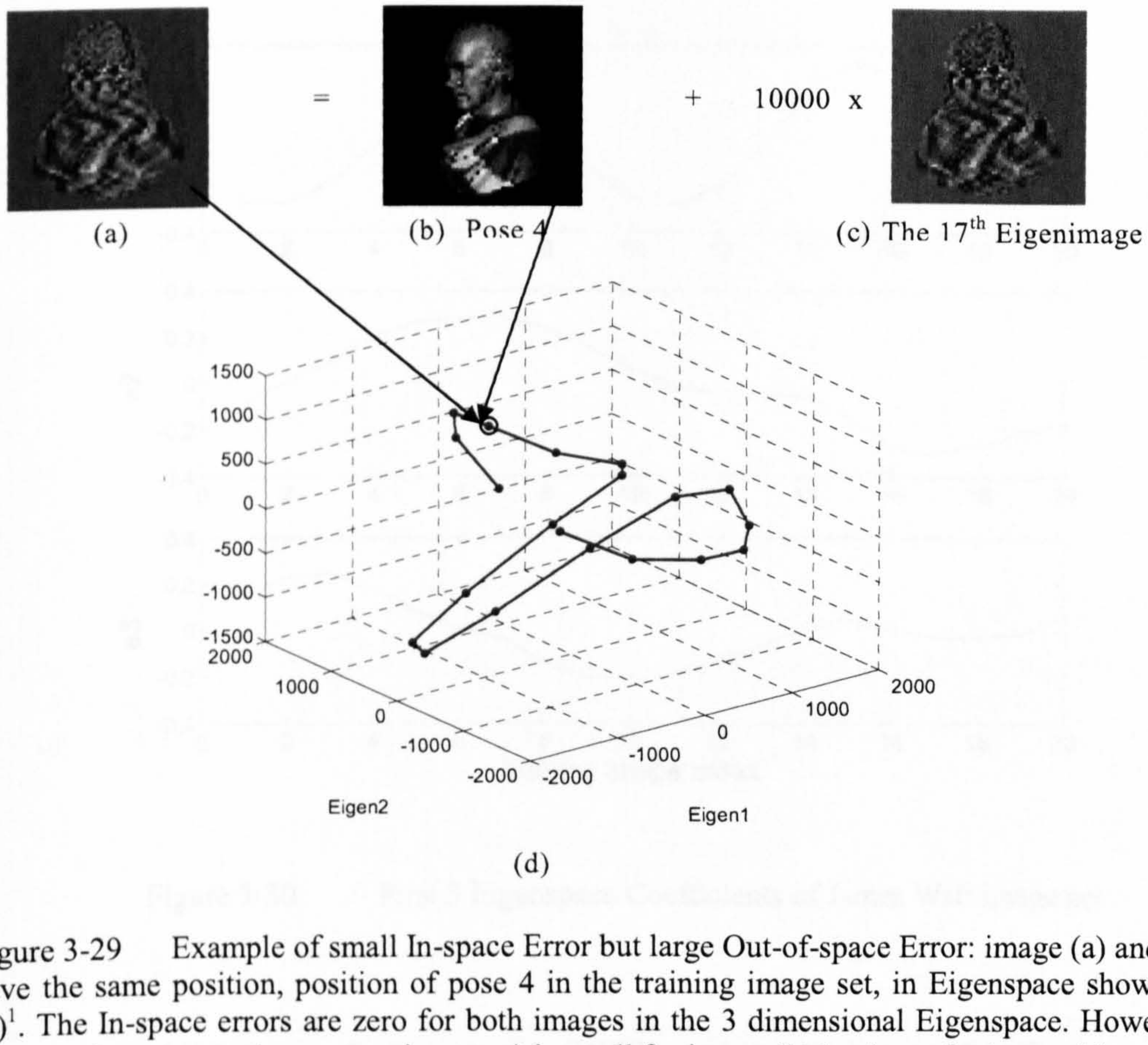
$$P(M \mid I) = \frac{P(M)P(I \mid M)}{P(I)} \tag{3-52}$$

$P(M)$ is the prior probability of the object, which can be seen as the probability that the object is present in the image before the image has been examined. $P(I/M)$ Is the likelihood that the image was generated by the model M. $P(I)$ is the probability of the image which we can ignore because it is constant across models. $P(M)$ can also be omitted if we assume a uniform prior because it will affect all of the models posterior's equally. Thus, only the likelihood term is left:

$$P(M \mid I) \propto P(I \mid M) \tag{3-53}$$

If we use several object Eigenspaces, a likelihood $P(I/M)$ (or the posterior probability $P(M/I)$) threshold can be defined for each object in the database to determine whether the object is present. If we use a universal Eigenspace, we choose the object with the highest likelihood as the object recognized.

This likelihood can be determined by two measures: the ***reconstruction error*** and the ***Eigenspace distance***. For images containing the objects used to build the Eigenspace, the Eigenspace distance should be small. However, there are images with their Eigenspace projection identical to the projection of a training image that looks nothing like the training image, e.g., one could take a training image and add one of the unused eigenimages times a large scalar to it. This is illustrated in Figure 3-29, where image (a) is a sum of pose 4 in the training set and 10000 times the 17th Eigenimage. Eigenspace (d) is the same one as in Figure 3-3. When projecting image (a) to the space formed by the first 3 dimensional Eigenvectors, the projection is identical to the projection of image (b), pose 4 in the training set. Here we only illustrate the first 3 dimensions, in fact, the Eigenspace projection for image (a) and (b) are identical in the Eigenspaces formed by the first 16



(a) (b) Pose 4 (c) The 17th Eigenimage

(d)

Figure 3-29 Example of small In-space Error but large Out-of-space Error: image (a) and (b) have the same position, position of pose 4 in the training image set, in Eigenspace shown in (d)[1]. The In-space errors are zero for both images in the 3 dimensional Eigenspace. However, the out-of-space error(reconstruction error) is small for image (b) but large for image (a).

Eigenvectors. This is because all the eigenvectors form an orthogonal basis, adding a multiple of an unused eigenvector to a training image will not affect the Eigenspace projection. This is a special case of the result we obtained in Equation (3-19). In Equation (3-19), we assumed that each image could be represented by a linear combination of all eigenvectors and thus the difference between two images can be represented as the distance of their projection in Eigenspace. In this case, the image contains a large portion of unused Eigenvector, which means it cannot be represented by a linear combination of the eigenvectors and the assumption no longer holds. This situation can be detected by checking the reconstruction error. If the reconstruction error is large, then the test image does not belong to the Eigenspace.

Likewise, the Eigenimage can be reconstructed with no error but it does not look similar to any images in the training set. In Moghaddam's work [14], they refer to the ***Eigenspace***

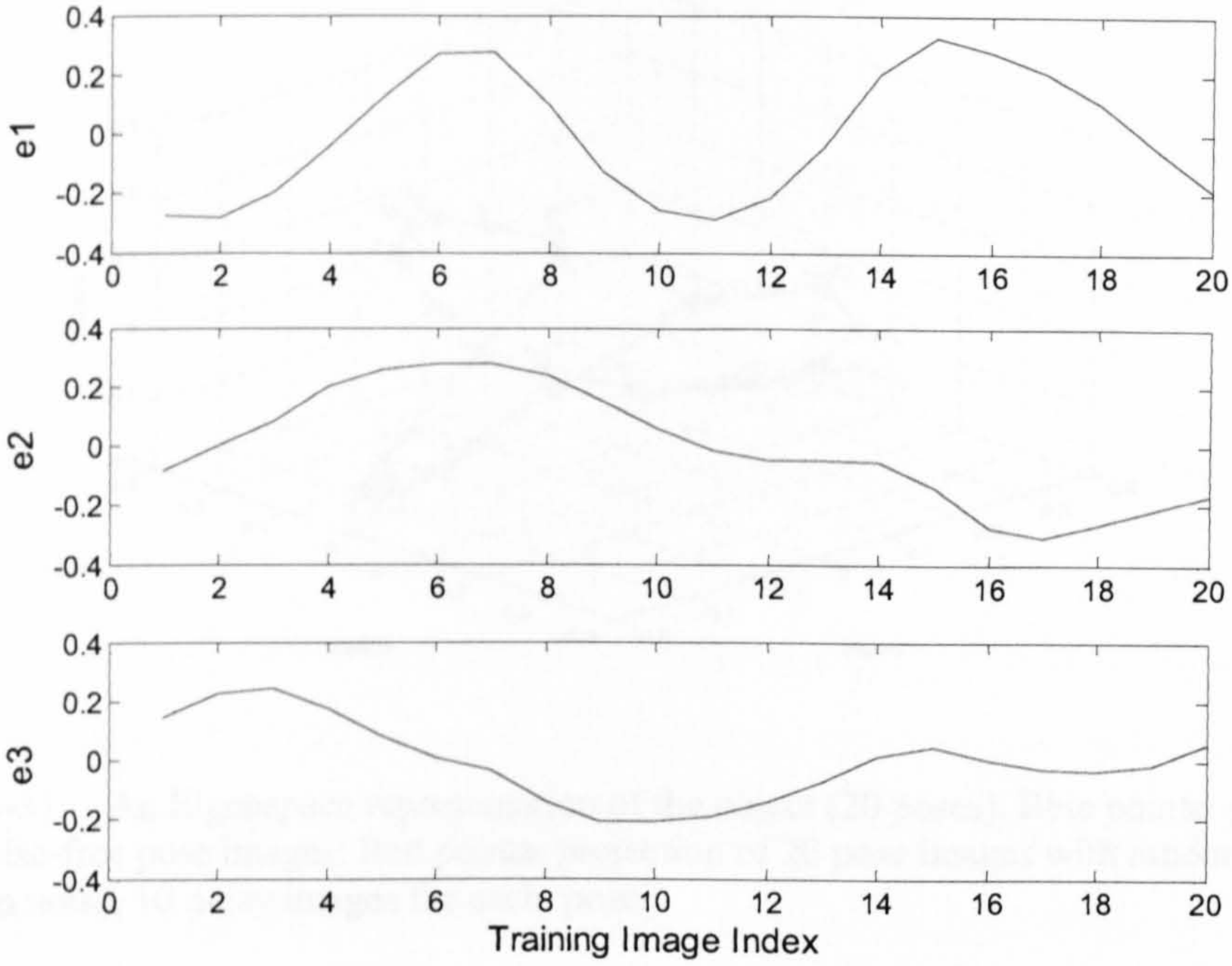


Figure 3-30 First 3 Eigenspace Coefficients of James Watt image set

distance as 'distance-in feature-space' (DIFS) and to the ***reconstruction error*** as 'distance-from-feature-space' (DFFS). Here we refer to the former as ***In-space error*** and the later as ***Out-of-space error***. The latter error is caused by the lack of eigenvectors. The former is caused by the Eigenspace method itself.

The ***In-space error*** reflects how probably the model represents the test image measured by the projection of that image in Eigenspace. We use $P(\vec{a} \mid M)$ to measure the likelihood that the coefficients $\vec{a}$ came from the model, where $\vec{a}$ is the Eigenspace coefficients obtained by projecting the test image into Eigenspace. The Out-of-space error reflects the probability of reconstruction error of each pixel in the test image. If we assume that pixel errors are independent, the total likelihood of all pixels from the model is the product of all the likelihoods of the individual pixels. If we use $P(I_i \mid \vec{a}, M)$ to represent the likelihood that the given pixel i came from the model, the total likelihood is the product of coefficient likelihood (measuring in-space error) and all the individual likelihoods (measuring out-of-space error):

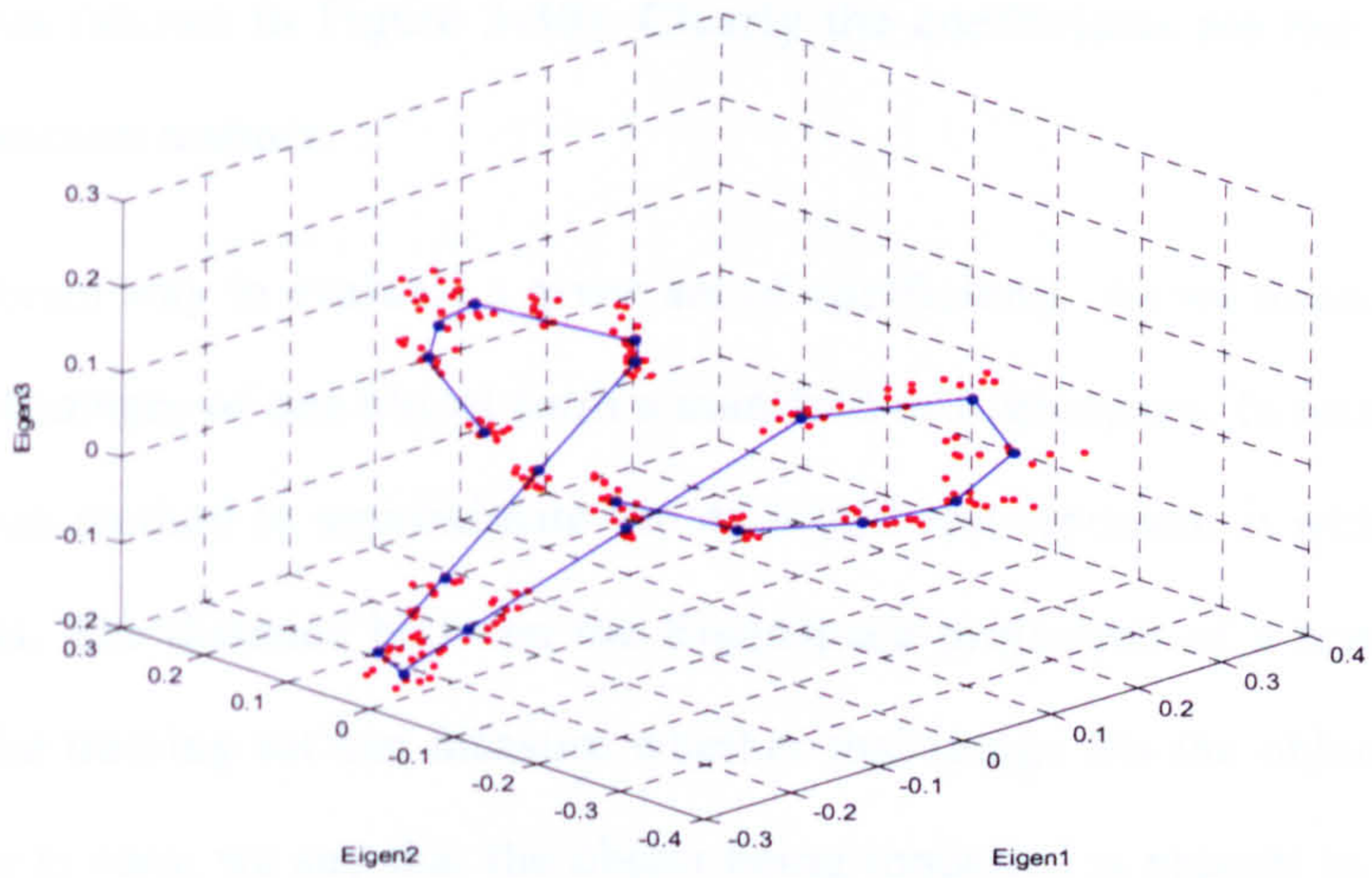


Figure 3-31 An Eigenspace representation of the object (20 poses). Blue points: projection of 20 noise-free pose images; Red points: projection of 20 pose images with random Gaussian noise, 10 noisy images for each pose.

$$P(I \mid M) = P(\vec{a} \mid M) \prod_{i=1}^{l} P(I_i \mid \vec{a}, M) \tag{3-54}$$

For simplicity, we convert the equation above to log domain to get

$$\log P(I \mid M) = \log P(\vec{a} \mid M) + \sum_{i=1}^{l} \log P(I_i \mid \vec{a}, M) \tag{3-55}$$

In fact, in most cases the pixel errors are not independent. Since it is difficult to calculate the correlations between them, we add a reduced factor γ to the term of pixel errors and this factor is determined experimentally. Then Equation (3-55) becomes:

$$\log P(I \mid M) = \log P(\vec{a} \mid M) + \gamma \sum_{i=1}^{l} \log P(I_i \mid \vec{a}, M) \tag{3-56}$$

3.5.2 In-Space Error

The in-space error is measured by the coefficient likelihood. If the model is build from a training set that contains similar views of images, e.g., faces or fingerprints, than the distribution of each of coefficients over the training set can be approximated as a Gaussian with zero mean [14]. However, in this thesis we are interested in 3D objects with multiple views, and the distribution of coefficients is not uni-modal. Using the image set shown in Figure 3-4 for example, we can plot the first three coefficients for each

training images (shown in Figure 3-30). Clearly the coefficients are not distributed in a uni-model Gaussian manner.

We use a different way to evaluate a given set of coefficients. As we discussed before, the coefficients of images of one object form a manifold in Eigenspace. In section 3.2, we use an interpolation method to approximate the manifold and represent it with a dense set of discrete points. The distance between the Eigenspace projection of a test image and the manifold of the training set can measure whether that image fits the object model or not. If the distance is zero, we say that the object being modelled is present in the image with no doubt. However, the projections of some test images are close to but not exactly on the manifold, though they contain the object being modelled. These images can be of slightly different poses from the training set, or with small in plane transformation, or with noises, etc.. All of these can move the projection of the test image away from the object manifold. However, we find that the effect of pose difference is different from other image transformations: the pose difference made the projection move along the manifold and other image transformations make the projection move away from the manifold without following any particular directions [9](see Figure 3-31). We have already modelled the pose variance by interpolation and now we are considering the effect of other image transformations.

As the distance becomes large, we can say that the probability of the object presence in the image gets smaller. Using this distance-to-manifold measure, we can formulate an approximation to $P(\vec{a} \mid M)$. For the projection of images that fit the object model, we approximate the density of distances by a Gaussian,

$$p(a \mid M) \approx N(d(a \mid T) \mid 0, \rho_d) \tag{3-57}$$

where T represents the object manifold and ρ_d is the variance estimate of the distances. The reasonable value of ρ_d is associated with the distance between training images. Half of the maximum of these closest distances, Δ, can be used as a distance threshold. We

[9] This does not include thermal variation.

would expect that any point that really belongs to the manifold would be within this distance. We can choose a ρ_d that reflects this idea, e.g., if we choose $\rho_d = (\Delta/3)^2$, then approximately 99.8% percent of the density will lie within Δ.

To convert this density function to a likelihood, we must integrate it over some range $[-\delta/2,\ \delta/2]$. This range will be quite small relative to Δ, so we assume that the density will be roughly constant over this small range. If we also assume δ is independent of $\vec{a}$, we can approximate $P(\vec{a} \mid M)$ as

$$\begin{aligned} P(\vec{a} \mid M) &\approx \int_{\delta/2}^{\delta/2} N\left(d(\vec{a},T)+x \mid 0,\rho_d\right)dx \\ &\approx N\left(d(\vec{a},T) \mid 0,\rho_d\right)\delta \end{aligned} \tag{3-58}$$

Since δ is constant, when we calculate $\log P(\vec{a} \mid M)$, the δ can be ignored.

The method we proposed to calculate the probability associated with the DIFS is alike the single Gaussian model in Moghaddam and Pentland's work [14]. However, in their work, the distance is measured between the projection of the input image and the origin, while in our work, the distance is between the projection of the input image and the manifold of the training object in Eigenspace.

3.5.3 Out-of-Space Error

The out-of-space error is measured by the pixel likelihoods. As discussed before, the out-of-space error is caused by a lack of eigenvectors: since we only use the first k eigenvectors, the first k eigenvalues determine the amount of variance in the training set accounted by the model. Thus if the total number of Eigenvector is n, we can use the remaining $n-k$ eigenvalues to estimate the variance in the training set not accounted for by the model. If we reconstruct the image using the coefficient $\vec{a}$, the variance of errors at each pixel can be approximated by Gaussian distribution with mean 0 and variance

$$\rho = \frac{1}{nl}\sum_{i=k+1}^{n}\lambda_i \tag{3-59}$$

where λ_i is the Eigenvalue and l is the number of pixels in the image.

While Eigenvalues indicate the amount of variance accounted for by each principal component, the eigenvectors themselves indicate where in the image the variance is accounted for. If the value of Eigenimage at a specific pixel is 0, then there is no variance at that pixel, and its eigenvalues should not be used in an estimate of the variance at that pixel. To account for this, the variance can be calculated for each pixel as

$$\rho_i = \frac{1}{n}\sum_{j=k+1}^{n}\left(e_{ji}\lambda_j\right) \tag{3-60}$$

where e_{ji} is the i th pixel in the j th Eigenimage. We can now approximate the density of errors as

$$p(I_i \mid \vec{a}, M) = N(I_i - U\vec{a} \mid 0, \rho_i) \tag{3-61}$$

where $N(\cdot \mid \mu, \rho)$ is a Gaussian density with mean μ and variance ρ. In Equation (3-61), U is the matrix of all used Eigenvectors, $U\vec{a}$ is the reconstructed image, and $(I_i - U\vec{a})$ is the reconstruction error. To convert this to a likelihood, we choose to integrate Equation (3-61) over the range of one grey level in a 255 gray level image. Using the midpoint rule and dividing the range into 2 intervals, we approximate the likelihood as:

$$P(I_i \mid \vec{a}, M) = 0.5 \cdot \left(p(I_i - 0.25 \mid \vec{a}, M) + p(I_i + 0.25 \mid \vec{a}, M)\right) \tag{3-62}$$

However, in our recognition algorithm, we normalize the energy of the test image to achieve a total unit energy by dividing the pixel values of the test image by a Energy normalization factor A, see Equation (3-2) and Section 3.1.1. So here the pixel values in the test image are not in the range [0 255] any more. To account for this, we divide the integration range by the Energy normalization factor A, and Equation (3-62) becomes:

$$P(I_i \mid \vec{a}, M) = \frac{0.5}{A} \cdot \left(p\left(I_i - \frac{0.25}{A} \mid \vec{a}, M \right) + p\left(I_i + \frac{0.25}{A} \mid \vec{a}, M \right) \right) \tag{3-63}$$

If we replace the right side of Equation (3-56) using Equation (3-58) and (3-63), we get the full expression of how to calculate the probability that the object being modelled is present in the test images.

3.5.4 Using Probabilistic Framework to solve small in plane transformation problem

The Eigenspace based method is based on good segmentation. However, even if a good segmentation is made, the object position in the test image may still be slightly different from the training image. For example, it may go several pixels up or down, left or right, or be rotated in the image plane. Even this is a small transformation, it will affect the result of the appearance based recognition to some extent.

To solve this problem, we propose to set an 'image window' in the test image. The 'image window' is of same size as the test image after scale normalization. The centre of the window starts at the centre of the test image and moves step by step up and down, left and right in a small scale, e.g., 1 pixel at each step. In each step, the window contains an image in which the object slightly changes position. The blank area is replaced by black pixels. Thus, in each step, the area in the 'image window' forms a new image. We use these new images as the test images and after the recognition procedure, each new test image is associated with an object or pose identity in a probability form. Thus, after the whole procedure, any original test image can end up at several objects or poses identity in a probability form. Finally, we choose the identity with the highest probability as the recognition result of the original test image.

3.6 Experiments

In this section, we test the basic Eigenspace based algorithm and its four improvements on both visible and infrared imagery. We use recognition rate as a measurement for the performance of the object recognition. The recognition rate is defined as the ratio of the number of successfully recognized images to the total number of test images.

3.6.1 Testing the basic algorithm on visible imagery

Objective:

The present experiments are designed to answer the questions:

(i) In the recognition processes, how many eigenvectors should we choose generally? How does this choice vary according to the size of the training set?

(ii) Recall that in the training process, we can build either a universal Eigenspace which contains the projection of all training objects or individual Eigenspaces each containing the projection of images of one object. One uncertainty surrounding the universal Eigenspace lies in the question: will it remain stable when the training set increases or decreases? In other words, does the size of the training set affect the performance of the universal Eigenspace?

Procedure:

We have used the COIL-100 [110] database in this experiment. COIL-100 is a database of images of 100 objects (see Figure 3-32). The objects were placed on a turntable which was rotated by 360 degrees to vary object pose with respect to a fixed camera. 72 poses were obtained at pose (azimuthally) intervals of 5 degrees.

Training image set and test images: In the present experiments, we use images of poses starting from 0 degrees and at intervals of 10 degrees for each object to be contained within the training images, and the other 36 images of each object as test images.

Figure 3-32 The Coil-100 Database

We start by randomly choosing two objects from the COIL-100 database. The 36x2 images generated are trained to form a two-object universal Eigenspace. In the recognition stage, we firstly use only one eigenvector, in other words, we treat the universal Eigenspace as a one-dimensional space and ignore other dimensions. We use the reminder of the 36x2 images of the two objects as test images. We increase the number of eigenvectors by one each time until we reach 20. At this point, we obtain 20 recognition rates corresponding to the 1 to 20 eigenvectors of a two-object universal Eigenspace. However, this is a special case. To generalize the result, we run the whole procedure 30 times and average the recognition rates. The process described above was then used for a total of 3, 4, ... 19 objects in the database.

Results:

The results of the experiments are shown in Figure 3-33 and Figure 3-34. As might be anticipated intuitively, Figure 3-33 shows that the overall performance of a smaller dataset (that is, containing a fewer number of objects) is better than that of a larger dataset under the same conditions. (For example, using one eigenvector, a 3 object dataset can achieve a recognition rate above 95% while an 18 object dataset only succeeds at around 85%. To achieve the same performance, the 18 object dataset needs to use more than 2 eigenvectors). However, for the large dataset, the recognition rate increases dramatically at the beginning as we increase the number of eigenvectors. The point at which the recognition rate becomes stable corresponds to the number of eigenvectors we should choose for a stable performance, e.g. for 3 object databases, we choose 5 eigenvectors for recognition and for 9 and 18 object databases, we choose 8 eigenvectors.

Our next goal is to examine whether the performance of a fixed dimensional Eigenspace is stable when the size of training set increases. As shown in Figure 3-34, a lower dimensional Eigenspace varies more than a higher dimensional Eigenspace as the size of training set changes.

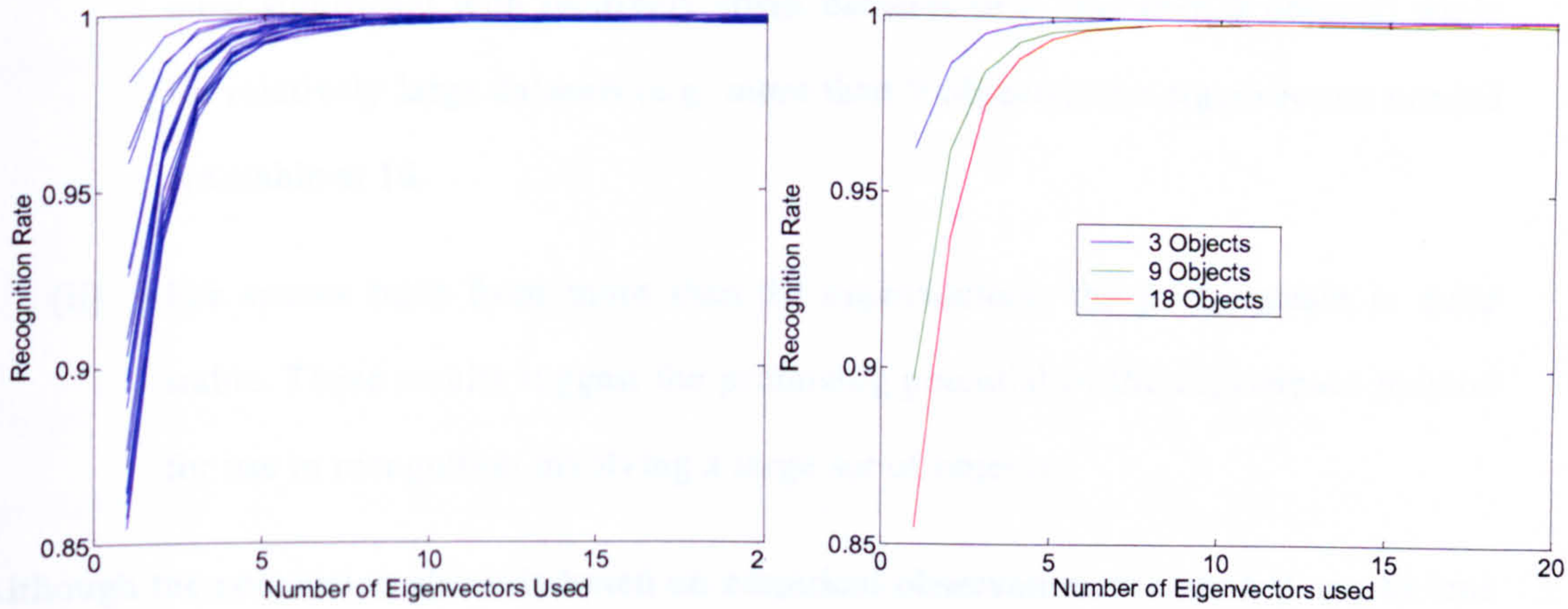


Figure 3-33 Each line in the plot shows the performance of a fixed size training set, e.g., the red line in the right plot shows the performance (recognition rate) variation when increasing the number of eigenvectors used in the recognition stage for an 18 object dataset. (Note that each line is an average of 30 trials.) the left plot shows the performance of 18 different sizes of training set. To provide clarity, the right plot shows only three different sizes.

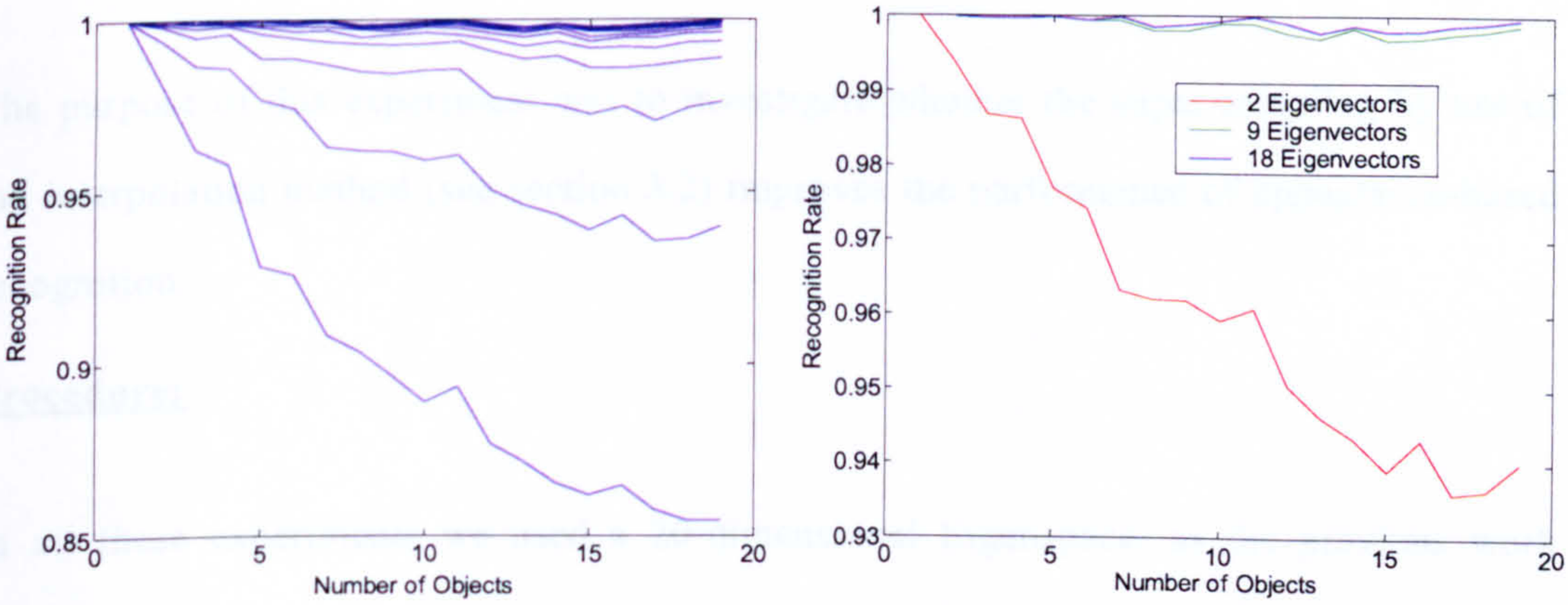


Figure 3-34 Each line in the plot shows the performance of a fixed dimensional Eigenspace as the size of training set increases, e.g., the red line in the right plot shows that the recognition rate drops as the number of objects in the training set increases from 2 to 19. (Note that each line is an average of 30 trials.) Left plot shows the performance of 20 different sizes of training set. To provide clarity for analysis, right plot shows only three different sizes.

Conclusion:

(i) Generally, we find that larger datasets require more eigenvectors. This trend is most significant with relatively small datasets (e.g. less than 9 objects) while for relatively large datasets (e.g. more than 9 objects), the eigenvectors needed are stable at 10.

(ii) For spaces built from more than 10 eigenvectors, the performance is quite stable. These results suggest the promising potential of the Eigenspace method for use in recognition involving a large set of objects.

Although the conclusion above is based on empirical observation, we think it can be true in general cases because first, the object class covers a large range and there are similar objects in each class; second, each recognition rate shown in Figure 3-33 and Figure 3-34 is an average of 30 different experiments on randomly chose objects.

3.6.2 Testing interpolation algorithm with noisy visible images

Objective:

The purpose of this experiment was to investigate whether the super-sampling by use of the interpolation method (see section 3.2) improves the performance of appearance-based recognition.

Procedure:

In all these experiments we used a 20-dimensional Eigenspace, as the previous work showed that the recognition rate tended to saturate before this point, at least for the limited object sets we have used to date. Of course, the illustrations show the first three dimensions of the manifold only. We compared the recognition accuracy and complexity with and without super-sampling the manifold in Eigenspace. We examined the effect of method of re-sampling the manifold when recognizing noisy images.

Training image set: we use objects from the Coil-20 dataset as our training set. In the coil-20 dataset, each object is represented by 72 images taken from pose (azimuthally) intervals of 5 degrees. We choose 36 poses (at intervals of 10 degrees) of each object as the training set.

Training Methods (see Figure 3-35):

Training method 1: build universal Eigenspace for all 36*20=720 training images.

Training method 2: build universal Eigenspace for all 36*20 training images and then resample each object manifold obtained in the training model to get 36*2*20=1440 points. (effectively super-sampling by a factor of two in azimuth).

Training method 3: build universal Eigenspace for all 36*20 training images and then resample each object manifold obtained in the training model to get 36*4*20=1440 points. (effectively super-sampling by a factor of four in azimuth).

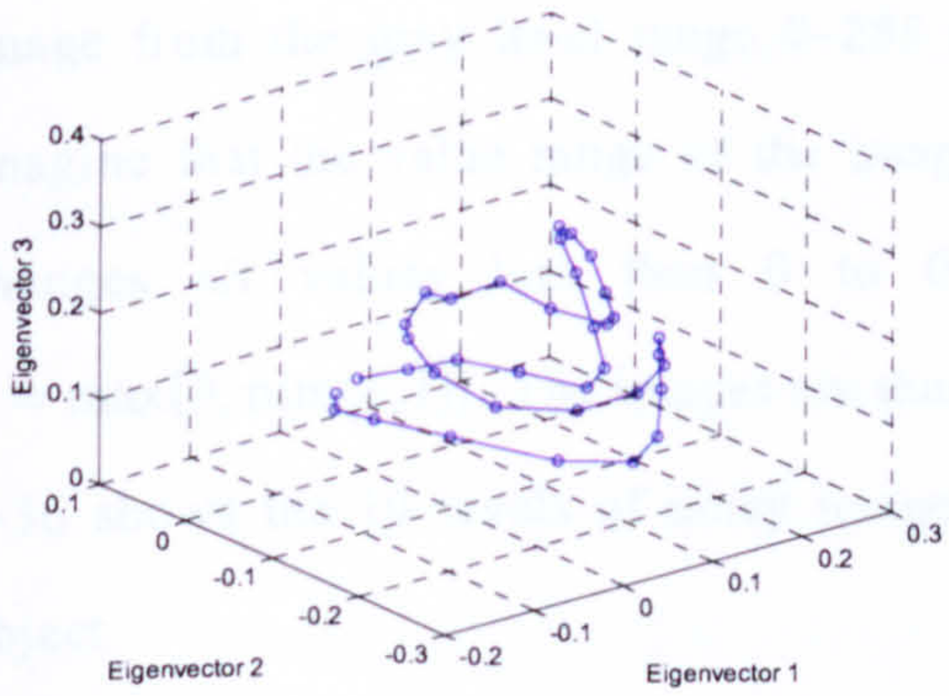


Figure 3-35 The upper left figure, upper right figure and the left figure show the resulting first 3D Eigenspace of the first object from Training method 1 (standard method) and Training method 2 (resample by a factor of 2) and Training method 3 (resample by a factor of 4)

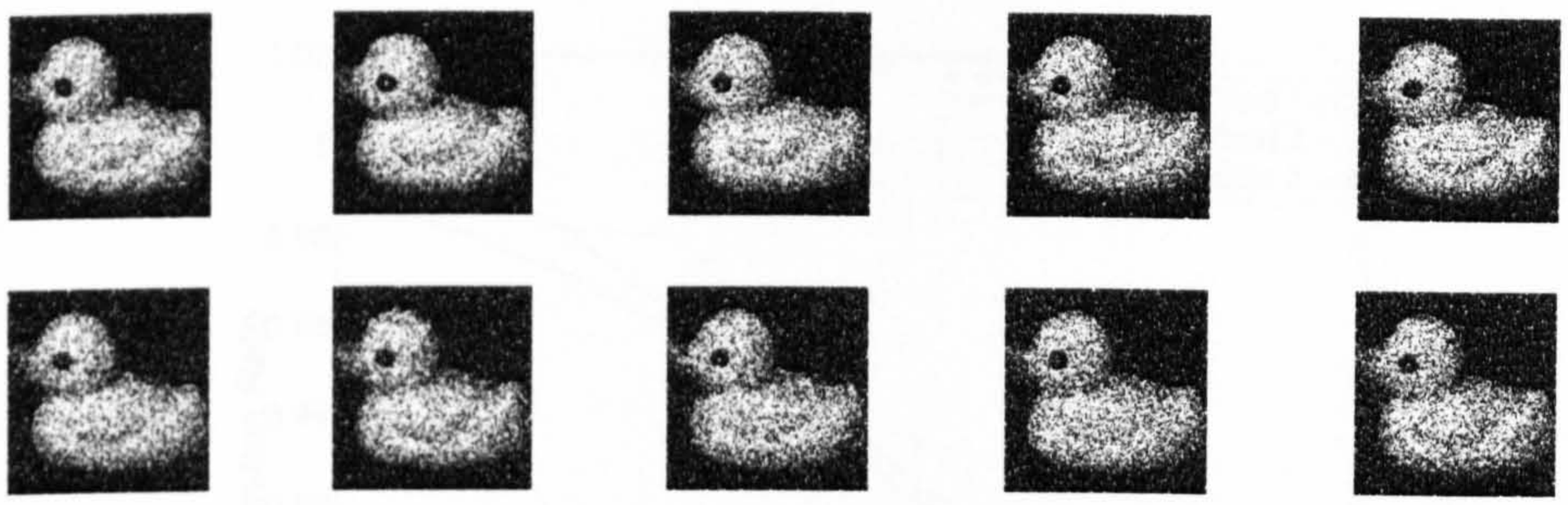


Figure 3-36 The first test image from the first object with 10 level of noises: upper line from left to right – noise level 1 to5, lower line from left to right noise level 6 to 10

Test images: In the description of the training image set, we said that we used half of the images in Coil-20 as the training images. We used the other half of the images (36*20=720) in Coil-20 to test our recognition method. In these experiments, we tested images with 10 levels of Gaussian noise. In practice, we use the Matlab function ***imnoise***. Images of the 10 noise levels are obtained by adding Gaussian noises $N(\mu, \sigma^2)$with a mean (μ) of '0' and variances (σ^2) of 0.10, 0.11, 0.12, 0.13, 0.14, 0.15, 0.16, 0.17, 0.18, 0.19 to all test images. Note that before adding the noise, the function normalize each image from the gray level range 0~255 to the range 0~1. After adding noise, we can imagine that the value range of the image is beyond [0 1]. The ***imnoise*** function then changes all values less than 0 to 0 and all values greater than 1 to 1 by $I = \max(0, \min(I, 1))$. The images are then restored to the gray level range 0~255. Figure 3-36 shows the 10 levels of noisy images of the first test image from the first testing object.

Results and Conclusion

Figure 3-37 summarizes the recognition results for the images with 10 different noise levels. We see that the recognition results form the two super-sampling methods, Training methods 2 and 3, are both better then the normal method without super-sampling. This means that the super sampling technique improves the recognition result when the input images were additive noise. For most noise levels, more dense super sampling (TM 3) is better than the less dense sampling (TM2). However, this is not always true for all noise

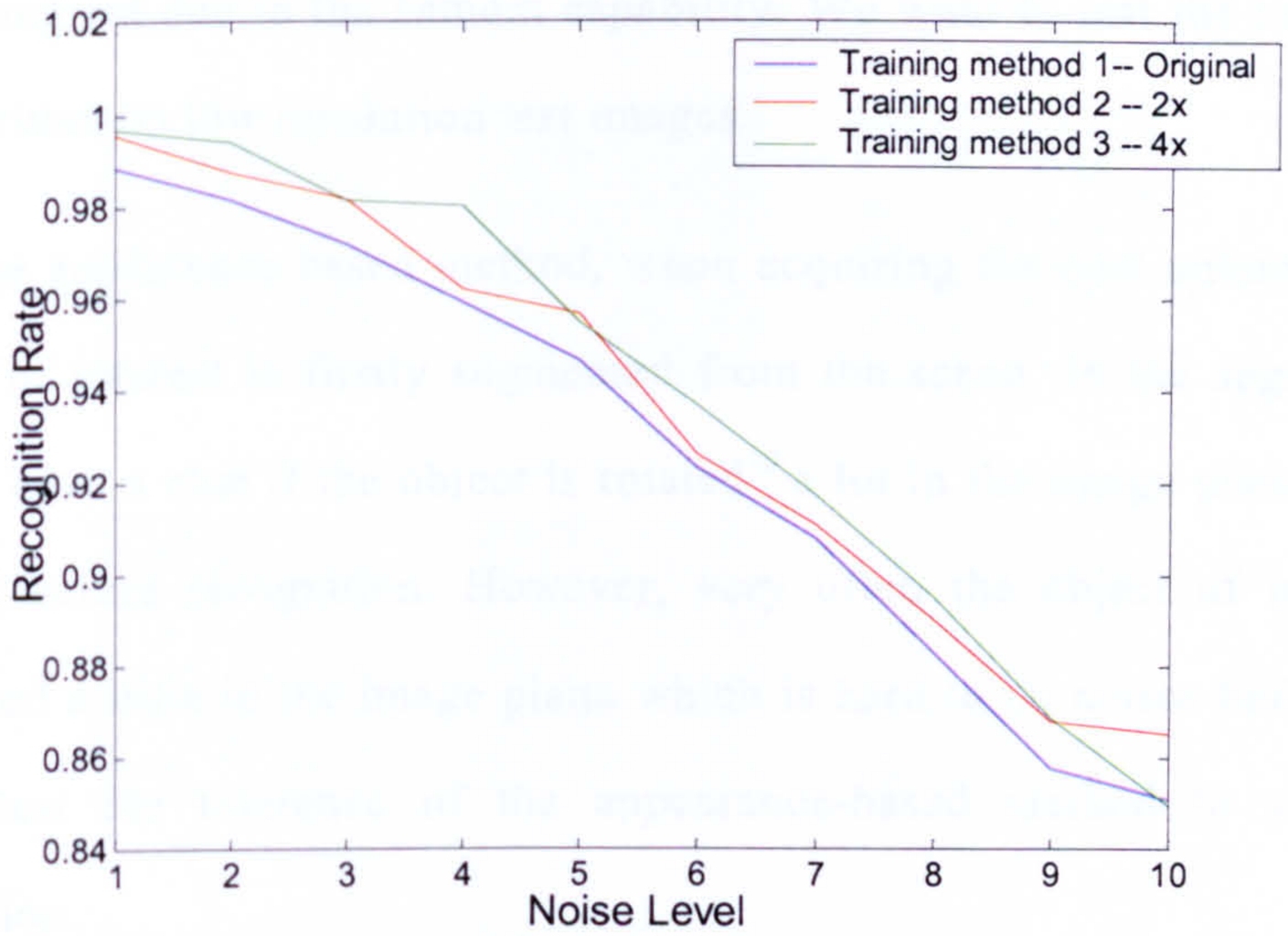


Figure 3-37 Recognition results of three training methods

levels, which means for certain applications, a certain sampling factor (e.g., 2 or 4 in this case) constrains the upper limit of this method. A bigger sampling factor does not always give a better result.

3.6.3 Testing the basic algorithm on LWIR imagery

Objective:

(i) In this experiment, we want to see how the Eigenspace-based recognition algorithm works on infrared images and how it is different from the visible images. In infrared imagery, we also want to examine the algorithm when test images are of different resolutions and rotated in the image plane.

(ii) For this experiment, we use images with 64*64 pixels as training images. For any new input image, we resize the image to 64*64 before going to the recognition process. In our database, all the input images are originally of size 200*200 which contains enough thermal information for recognition. However, in practice, in some applications, we cannot get enough thermal information, in other words, the input image may be of lower resolution than the images in the

training set due to the camera capability. We want to test the tolerance of the algorithm on low resolution test images.

(iii) In the appearance based method, when acquiring the new unknown scene, the part of interest is firstly segmented from the scene. In the segmentation, we may have a clue if the object is rotated[10] a lot in the image plane and rotate it back before recognition. However, very often the object of interest is just rotated a little in the image plane which is hard to be notice beforehand. Here we test the tolerance of the appearance-based method to small in-plane rotation.

Procedure:

In our database, we include 11 vehicles as shown in Figure 3-38 (a)-- 3 cars, 3 tanks, 2 landrovers, 1 truck, 1 helicopter and 1 scud. Each object is represented by 337 images from viewpoints spaced equally over the upper viewing hemisphere. The viewing positions (see Figure 3-38 (b)) were obtained by subdividing the faces of an Icosahedron to the third recursion level. The CameoSim package [118] [119] [120] (see section 5.1 for a description of the package) was used by Dr. Matt Kitchin to simulate our infrared images on request.

Training image set and test images: The method is similar to the leave-one-out strategy -- In the training process, leave one pose of each object out and use the other poses as the training set. In the testing process, use that one pose as the testing pose and see which object the pose belongs to and to which pose the testing pose is closest. This method guarantees that the test image is an unknown image. In our experiment, to save the computer's memory and computing time, we leave 6 poses of each object out at one time, using the other poses as the training set, and testing these 6 poses of each object in the recognition stage. We call it the leave-six-out strategy. The 6 poses are randomly chosen.

---

[10] Compared with the image orientation in the dataset.

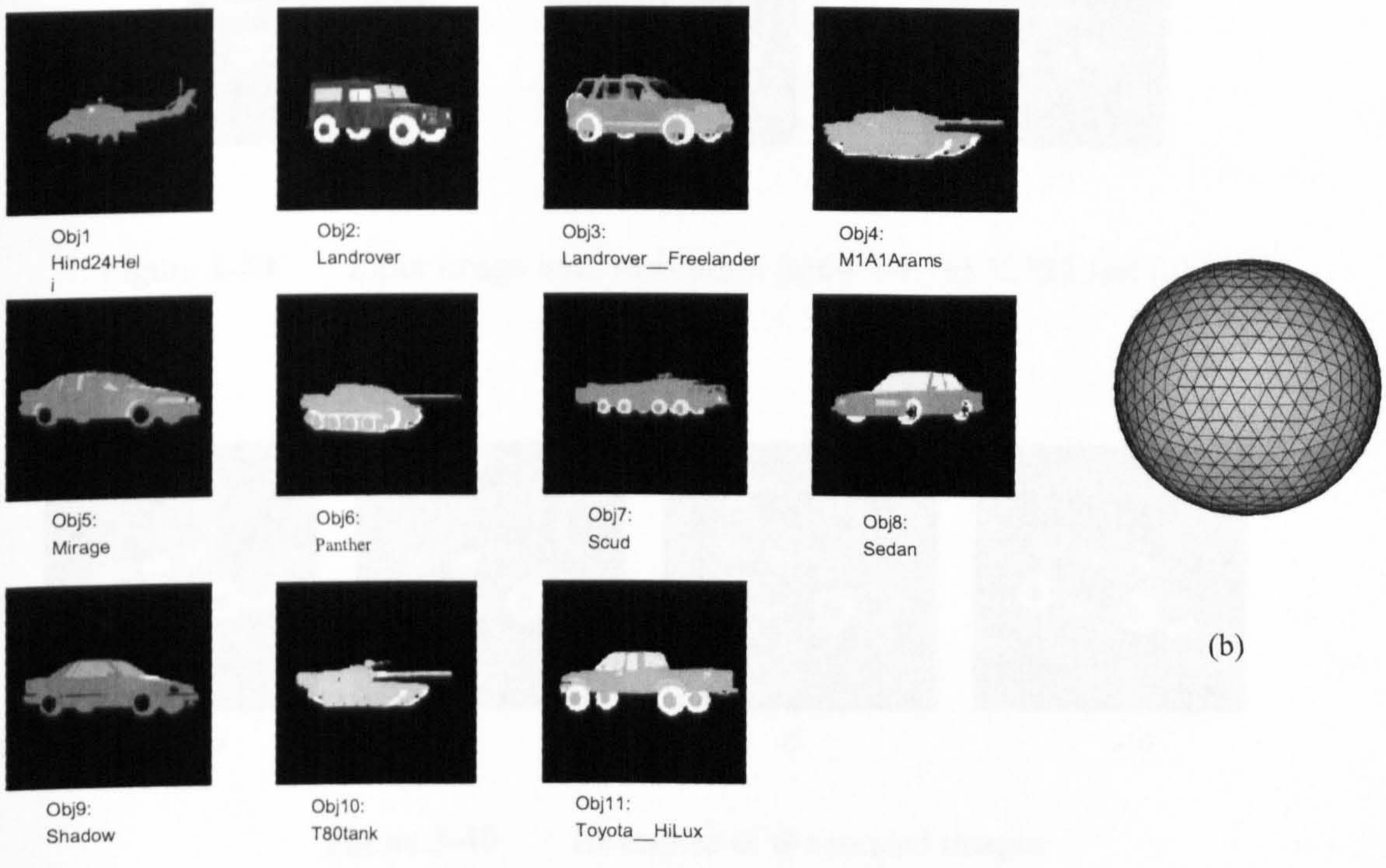
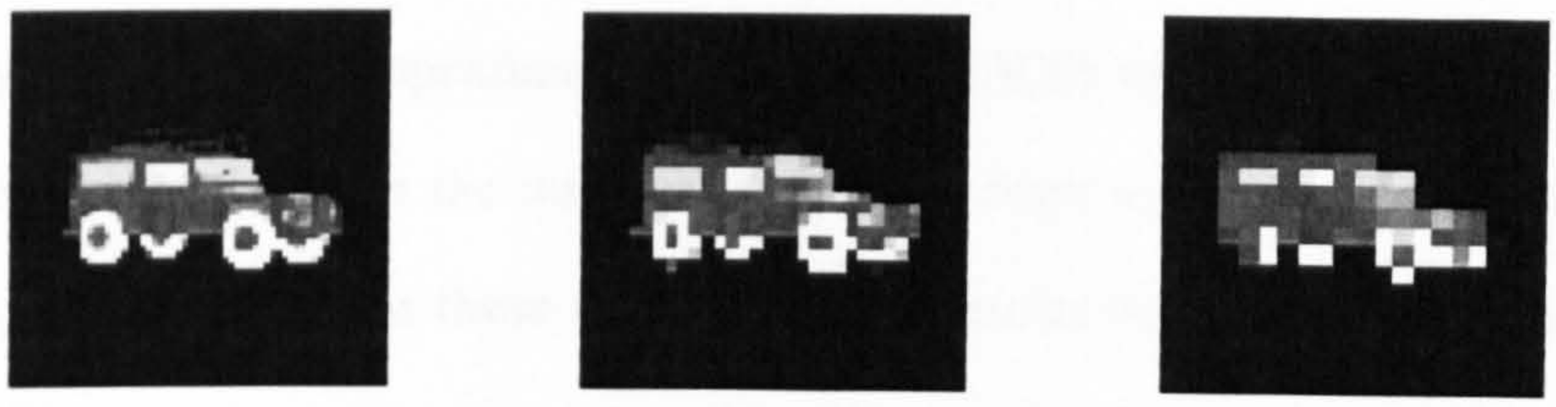


Figure 3-38 (a) Pose-1, T-1 of each object in database (b) Third recursion level Icosahedrons. The viewpoints of the images are the vertices in the upper sphere.

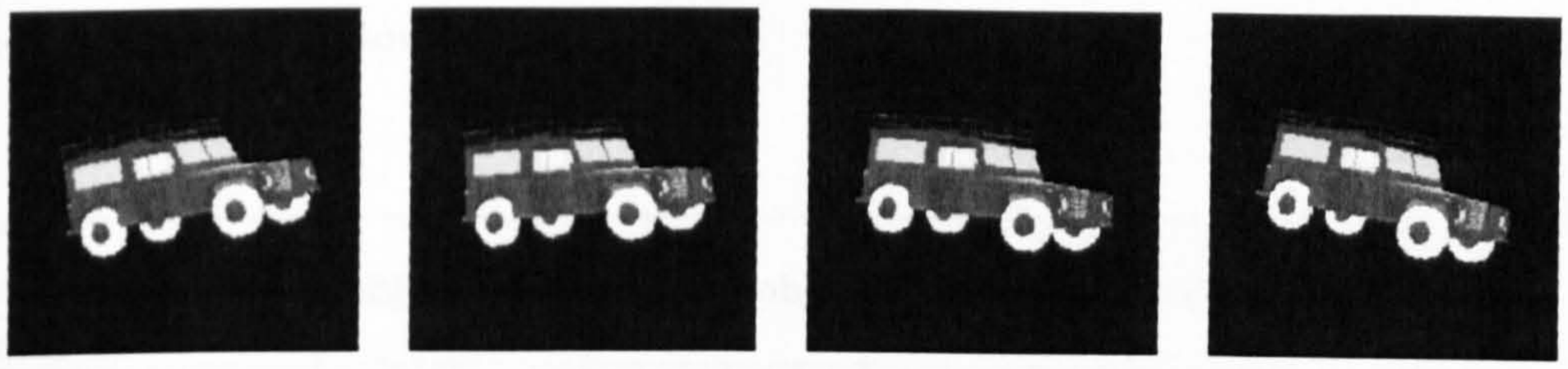
Fourteen sets of ***leave-six-out*** tests were done. The different poses that were randomly selected are listed below.

TEST SET	POSES					
1	265	281	250	81	157	195
2	215	190	60	211	329	10
3	123	107	285	15	158	79
4	278	50	316	309	73	137
5	16	178	27	149	108	231
6	320	291	159	318	133	243
7	293	17	126	302	120	8
8	20	113	156	85	201	312
9	30	244	55	70	86	9
10	7	184	234	1	84	115
11	274	2	163	305	62	221
12	45	272	111	258	251	187
13	83	59	275	58	24	71
14	138	109	267	18	199	77



(a) (b) (c)

Figure 3-39 Input image with resolution (a)64*64, (b) 32*32 and (c) 20*20



Rotate Degree: 10 5 -5 -10

Figure 3-40 Examples of the rotated images

The Universal Eigenspace method is used. We test using 10 different number of Eigenvectors: 10, 20, ... 100 and an image resolution of 64x64.

Test images of Different Resolution: We use images of 20x20 and 32x32 (see **Figure 3-39**) using 20 eigenvectors.

Testing Rotated Images: we obtain rotated images by rotating the object in the image of -10, -5, 5 and 10 degrees counter-clockwise (see Figure 3-40).

Results and Discussion:

Result of using different number of Eigenvectors: Results of using 20 Eigenvectors are shown in the following tables. Figure 3-41 shows a summery of all testing number of eigenvectors.

We see that object 5, 8, and 9 are most difficult to identify: they are easily confused with each other. From Figure 3-38, we see that objects 5, 8, and 9 are cars of a different

marque. They are similar in appearance, especially in LWIR imagery, when the texture of the car is more blurred. When the number of Eigenvectors used is increased from 10 to 100, the recognition results for these three objects becomes better. This is because in the sequence of Eigenvectors, the ones more heavily weighted count the main difference between training images and the back ones describe more detailed difference between the training images. The difference between these three objects is minus, so the Eigenvectors in the back matters. However, even using 100 eigenvectors, the recognition rate of these three objects is still below 70%.

Test image / Recognize as	obj1	obj2	obj3	obj4	obj5	obj6	obj7	obj8	obj9	obj10	obj11
obj1	84	0	0	0	0	0	0	0	0	0	0
obj2	0	84	0	0	0	0	0	0	0	0	0
obj3	0	0	83	1	0	0	0	1	0	0	0
obj4	0	0	0	80	0	1	0	0	0	1	6
obj5	0	0	0	0	46	0	0	31	16	0	3
obj6	0	0	0	0	0	77	0	0	0	7	0
obj7	0	0	0	0	0	0	81	0	0	4	0
obj8	0	0	1	0	26	0	0	32	26	0	1
obj9	0	0	0	0	10	0	0	47	40	0	5
obj10	0	0	0	0	0	6	3	0	0	81	0
obj11	0	0	0	3	2	0	0	0	2	0	76
Recognition Rate (%)	100.	100.	98.8	95.2	54.8	91.7	96.4	38.1	47.6	96.4	90.5
Number of Eigenvectors: 20											

The lower figure in Figure 3-41 shows an average recognition rate of the 11 objects (blue line). Comparing this result with the experimental result with visible images shown in Figure 3-33, we see that for the visible images of around 10 objects, the recognition rate almost reaches 100% with more than 10 eigenvectors, while with infrared images, the recognition rate is below 90% even using 100 eigenvectors. We would argue that this is because they are using totally different set of training and testing objects, e.g., in infrared case, we use three objects that have similar appearance in infrared imagery. The red line in the lower figure in Figure 3-41 shows an average recognition rate of the objects

without the 3 cars. We see that without including the three cars, the recognition rate improves by more than 10% and becomes stable at using 20 eigenvectors. However, it's still not as good as the result using visible images.

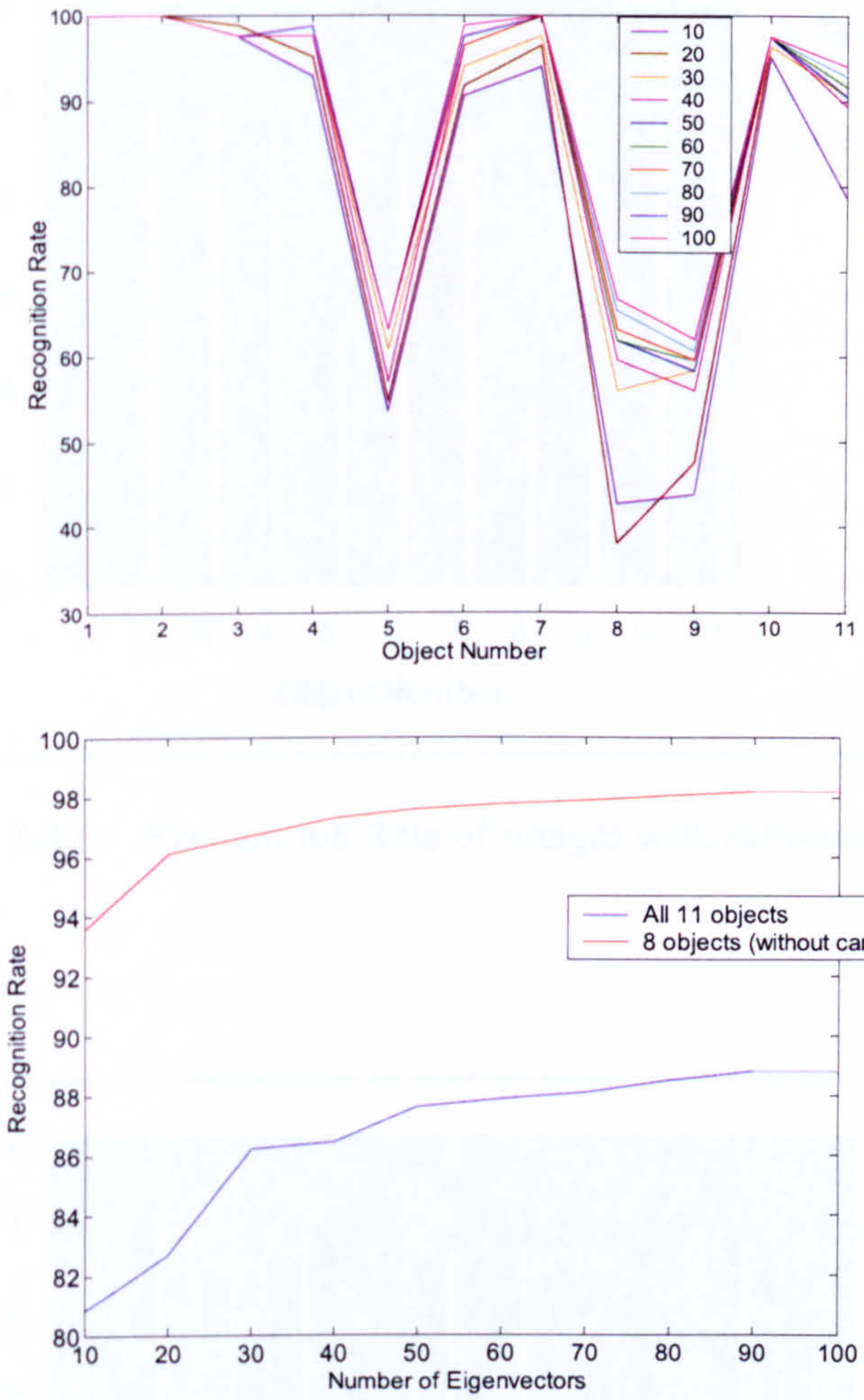


Figure 3-41 Results of testing the effect of different number of Eigenvectors used on basis Eigenspace method

Figure 3-42 shows the recognition results of the three different resolution images. We see that images with lower resolution are not necessarily harder to recognize. In other words, the appearance-based method deals well with low resolution images.

Figure 3-43 shows the recognition results of rotated images within the image plane of different angles. We see that the rotation does not affect the recognition rate very much.

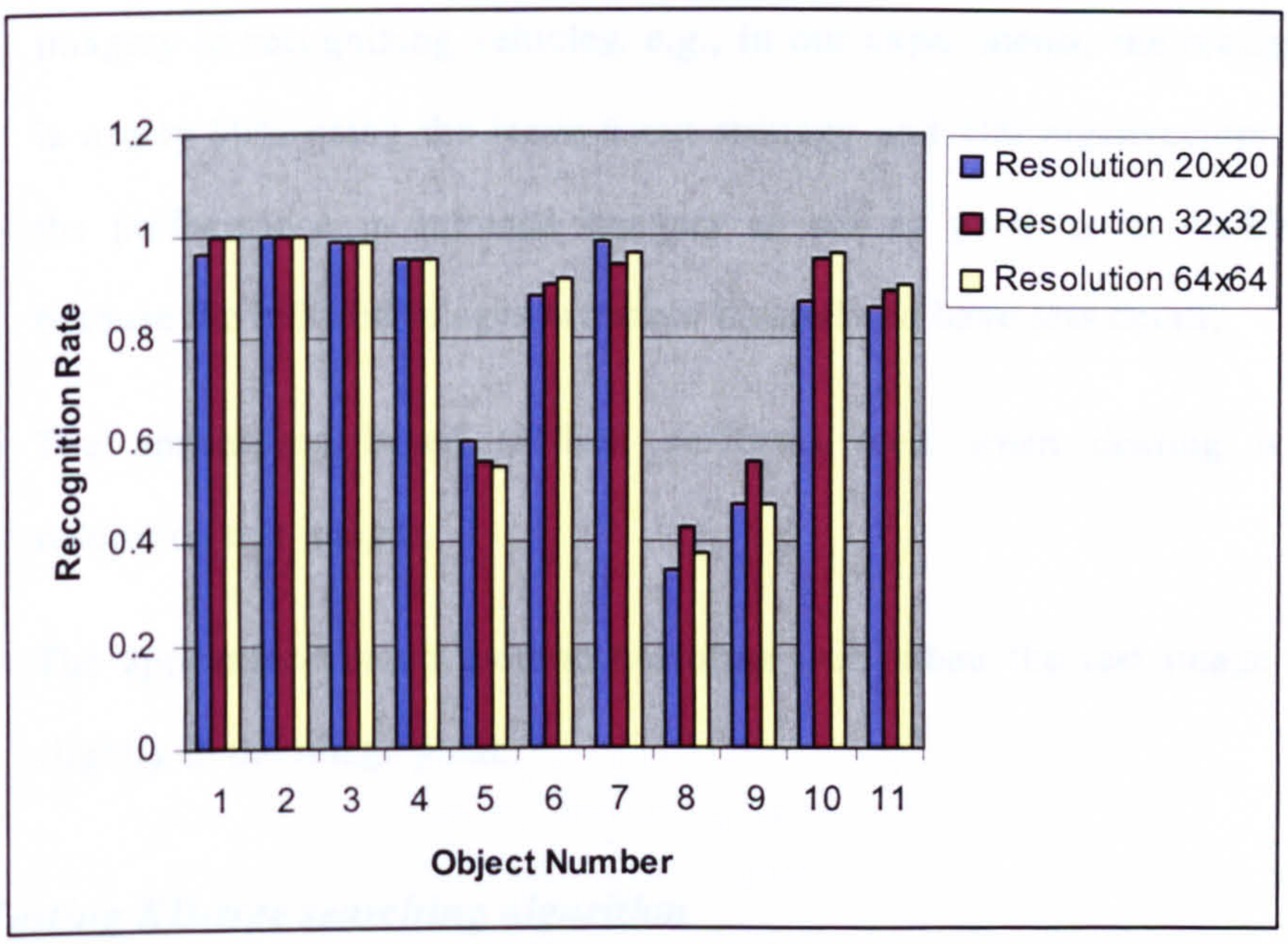


Figure 3-42 Recognition Rate of images with different resolutions

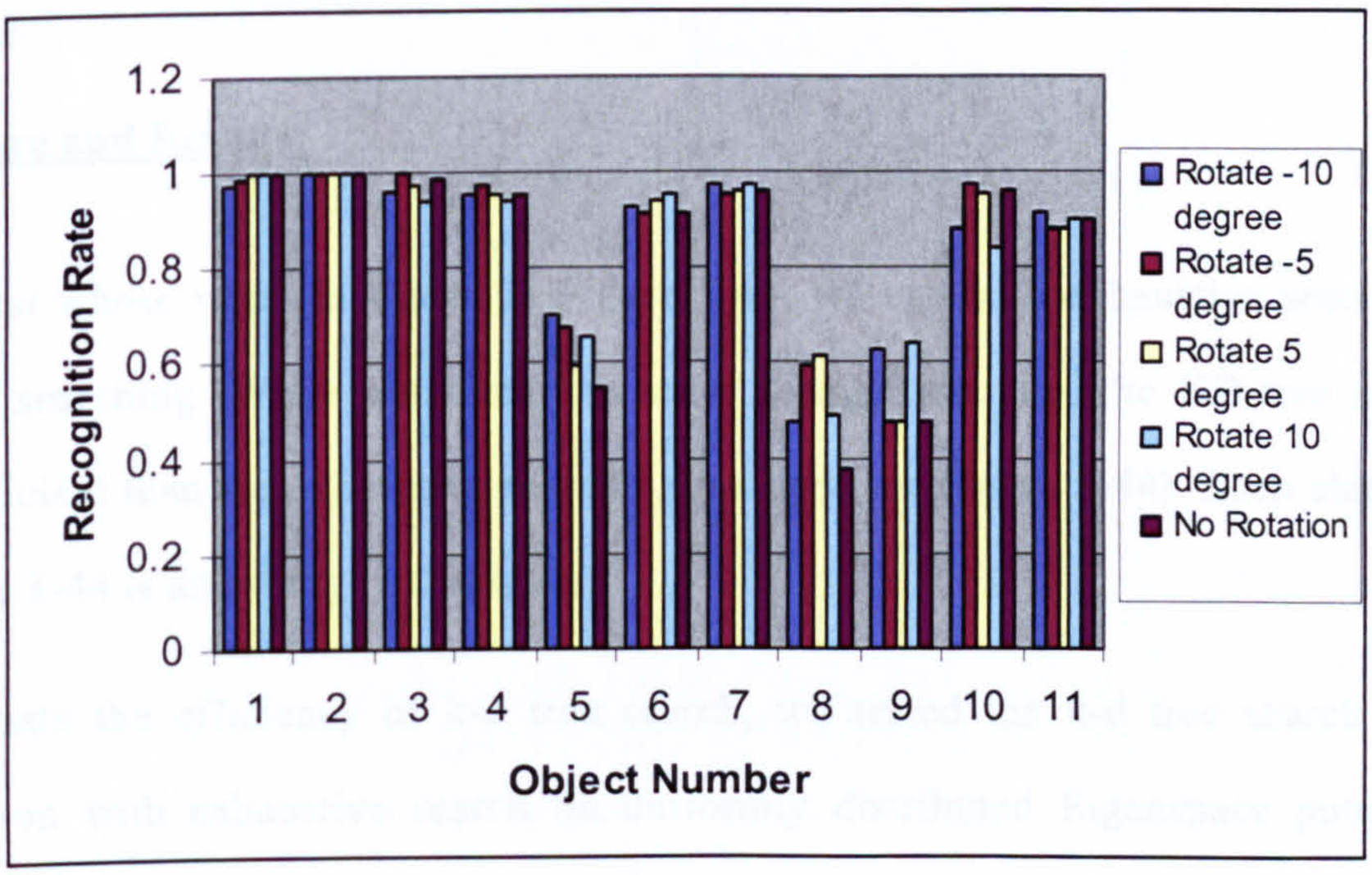


Figure 3-43 Recognition results of the rotated images

Conclusion

(i) The appearance based method can be applied in object recognition in infrared imagery in recognizing vehicles, e.g., in our experiments, the recognition rate is nearly 90% using the leave-6-out strategy and 100 eigenvectors. However, the performance in infrared imagery is not as good as in visible imagery because the infrared images are more blurred and have less detail;

(ii) The appearance based method performs well when dealing with lower resolution test images;

(iii) The appearance based method performs well when the test image is rotated slightly in the image plane.

3.6.4 Testing KD-tree searching algorithm

Objective:

In this experiment, we compare the efficiency of exhaustive searching and KD-tree searching.

Procedure and Results:

In the test whose result is shown in Figure 3-41, we use both exhaustive searching and KD-tree searching. While obtaining the same recognition rate, the KD-tree method is more efficient than the exhaustive searching method (see Figure 3-44). Each elapsed time in Figure 3-44 is an average of 60 trials.

To evaluate the efficiency of k-d tree search, we tested the k-d tree search again in comparison with exhaustive search on uniformly distributed Eigenspace points. From Figure 3-45, we can see that k-d tree method is much faster than the exhaustive search. Noting that the x-axis in each graph is a power series, the exhaustive search procedure is linear in the number of points, n, but the kd-tree is of complexity $\log_2 n$, as anticipated from the basic theory. Hence, for large object libraries and/or large numbers of variables

in the viewing conditions the k-d tree search should accelerate the search procedure significantly.

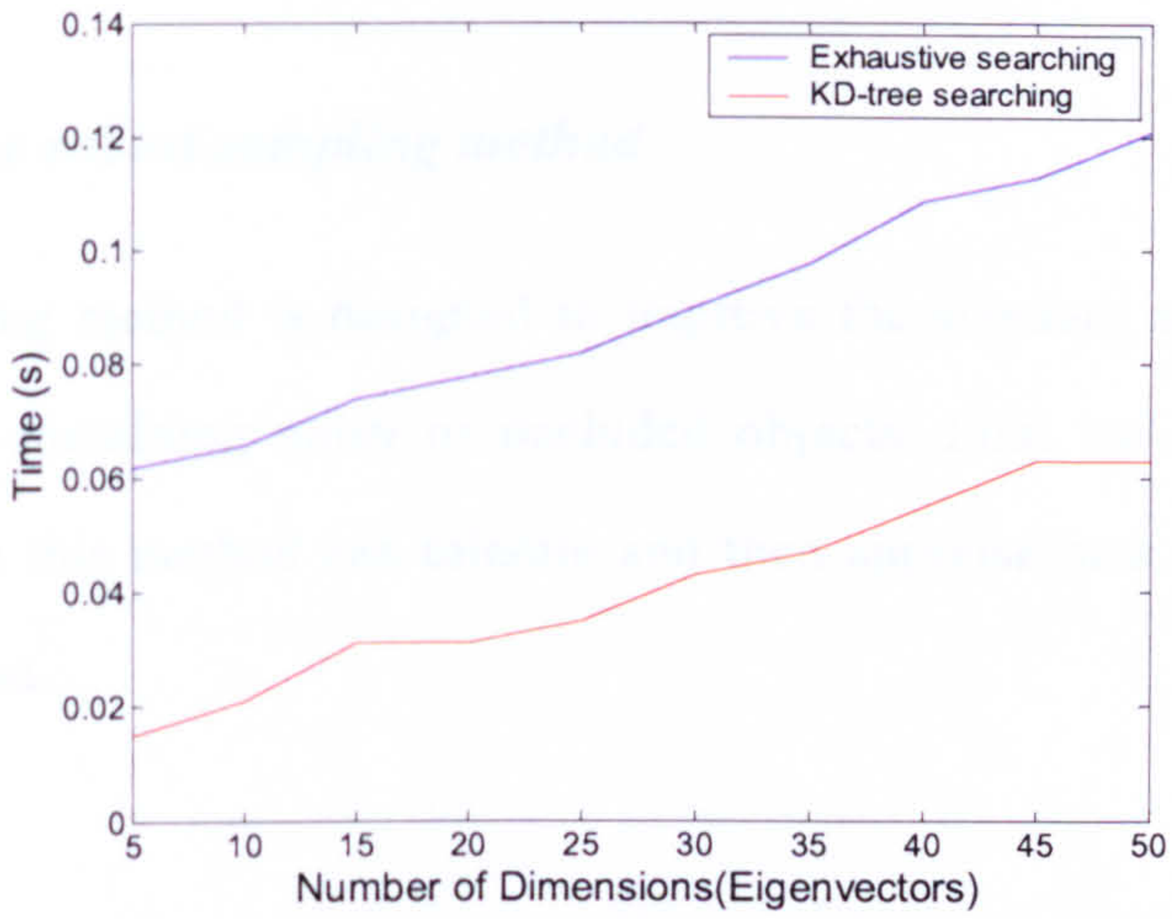


Figure 3-44 Comparison between KD-tree searching and Exhaustive searching

Figure 3-45 The execution time plotted against the number of Eigenspace points for two different values of the number of dimensions, k .

Conclusion

(i) KD-tree searching is much more efficient than exhaustive searching;

(ii) The elapsed time of KD-tree searching is independent of dimensionality.

3.6.5 Testing the robust sampling method

The robust sampling method is designed to improve the standard method in that it may recognize images containing noisy or occluded objects. First we want to examine the noise levels which this method can tolerate and then appraise how much it improves on the standard method.

Experiment 1

Objective:

The goal of this experiment was evaluate how much the robust sampling method improves the performance in recognizing noisy images, as measured by ***projection distance***.

Procedure:

We use the training image set as shown in Figure 4-4, 20 poses from one object, to form a 9-dimensional Eigenspace. Then we choose pose 1 and add different densities of 'salt and pepper' noise to generate the test images. In the program, we use Matlab function IMNOISE(I, 'salt & pepper', D), where I is the original image and D is the noise density. For example, D=0.1 means that 10% present of the image area is replaced by 'salt and pepper' noise. We tested images with noise density from 0 to 90% and at 0.25% interval.

The principle of the robust method is to select the 'undamaged' pixels in the image and minimise the influence of the noisy pixels. If we successfully select the 'undamaged' pixels, the reconstruction error should be small and the projection of the testing noised images in Eigenspace should be close to the projection of original image. Thus we use the distance between the projection of the noisy test image and projection of the original

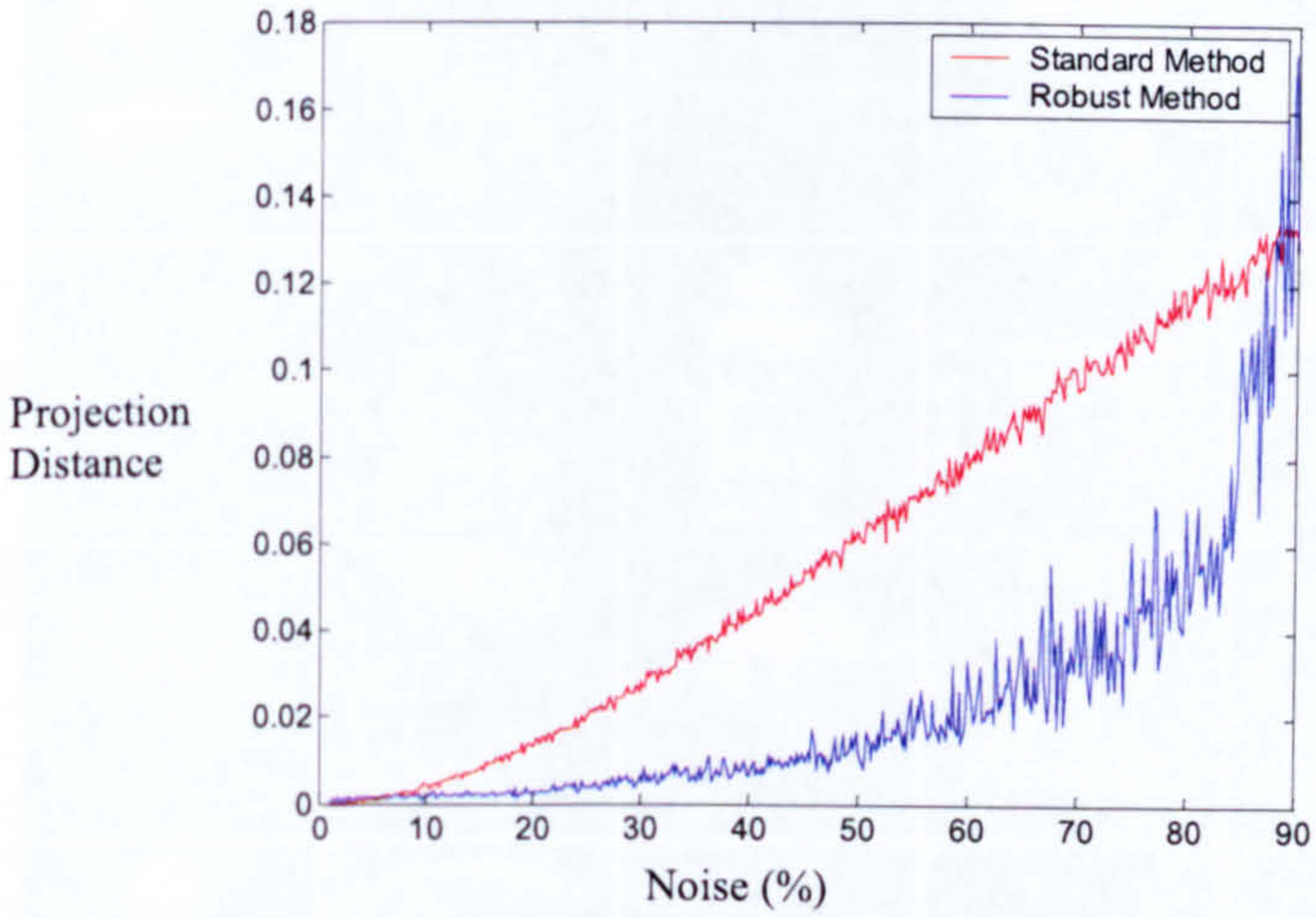


Figure 3-46 Monte Carlo simulation showing the Projection Distance as a function of noise for the robust sampling method

image to measure the robustness of the sub-sampling method. We call this the ***projection distance***. The smaller the projection distance, the more robustness the recognition method.

We tested the robustness using a simulated Monte Carlo approach where the noisy pixels are selected randomly and at each noise level, 30 different test images are tested. Thus the result of each noise level is an average of 30 trials.

Result and Conclusion:

Figure 3-46 shows the Projection Distance as a function of noise for both the robust sampling method and the standard method. We see that as the noise level increase, the projection distance for the standard method increases linearly. While using the robust method, the projection distance is low and stable up to 50% noise density. The robust method reduces the ***projection distance*** caused by image noises. For example, using the robust method, the result of testing noisy images of level 40% is comparable to noise level of 15% using standard method.

Figure 3-47 Coil-20 database

Experiment 2

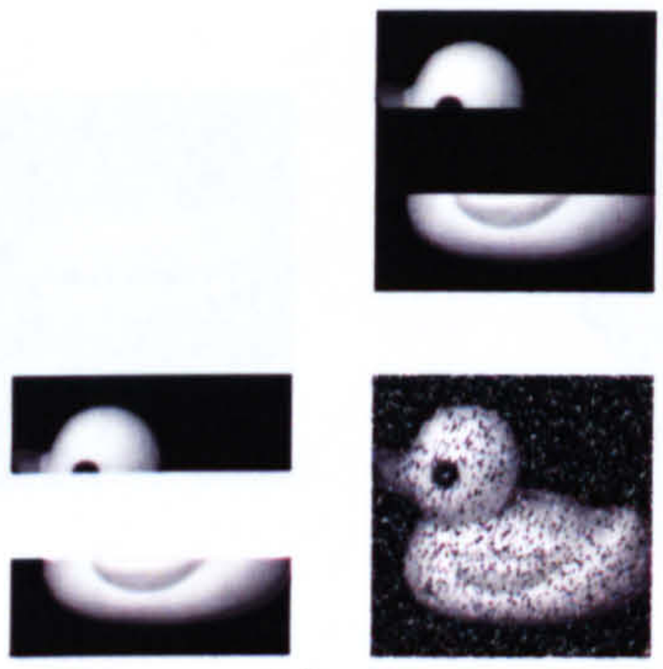
Objective:

The goal of this experiment is to compare the robust method with the standard method measured by ***recognition rate*** in object classification.

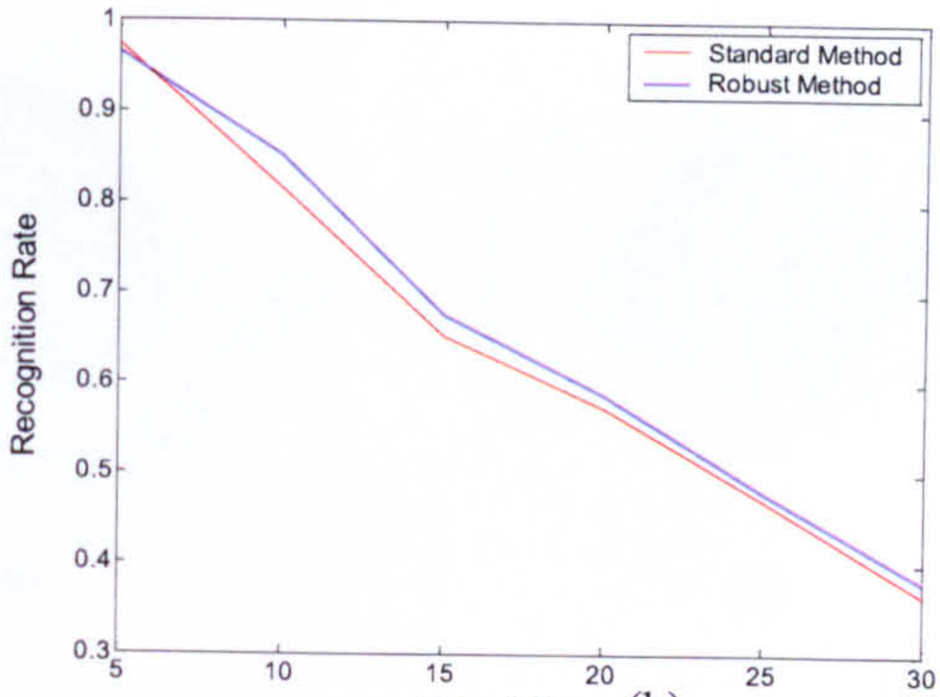
Procedure:

In this experiment we use 20 objects from the COIL-20 database (see Figure 3-47). In the database, each object is represented by 72 images from 72 poses obtained at pose (azimuthally) intervals of 5 degree. We used 36 poses starting from 0 degree and at intervals of 10 degree of each object as training set. Ten eigenvectors are used for both methods. We used the recognition rate to measure the performance.

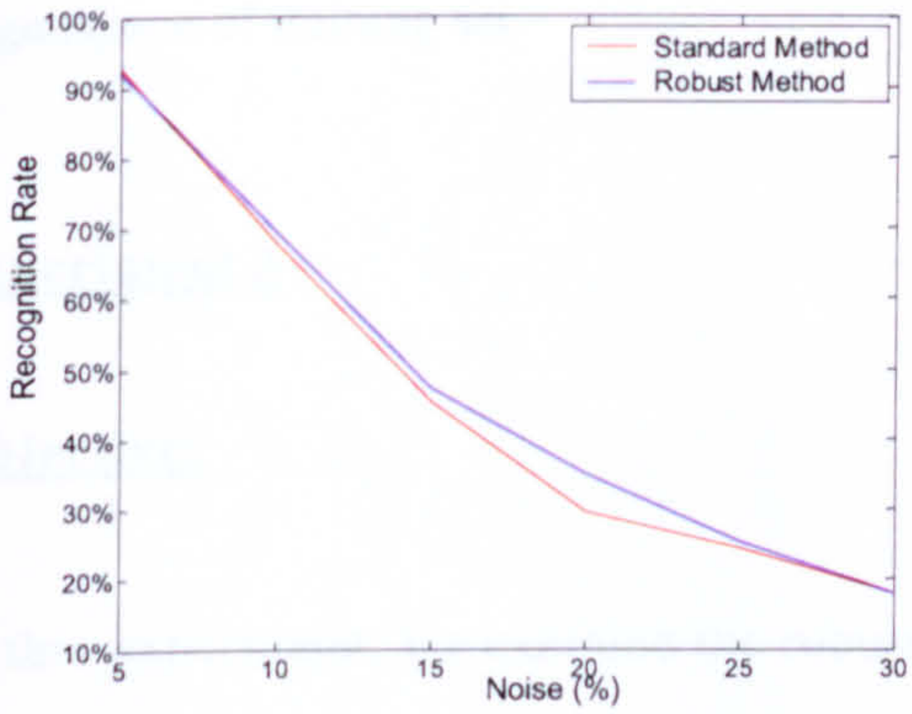
We added three types of noise/occlusion to the image: Salt & Pepper noise, black occlusion and white occlusion. Figure 3-48(a) shows the effect of noise and occlusion.



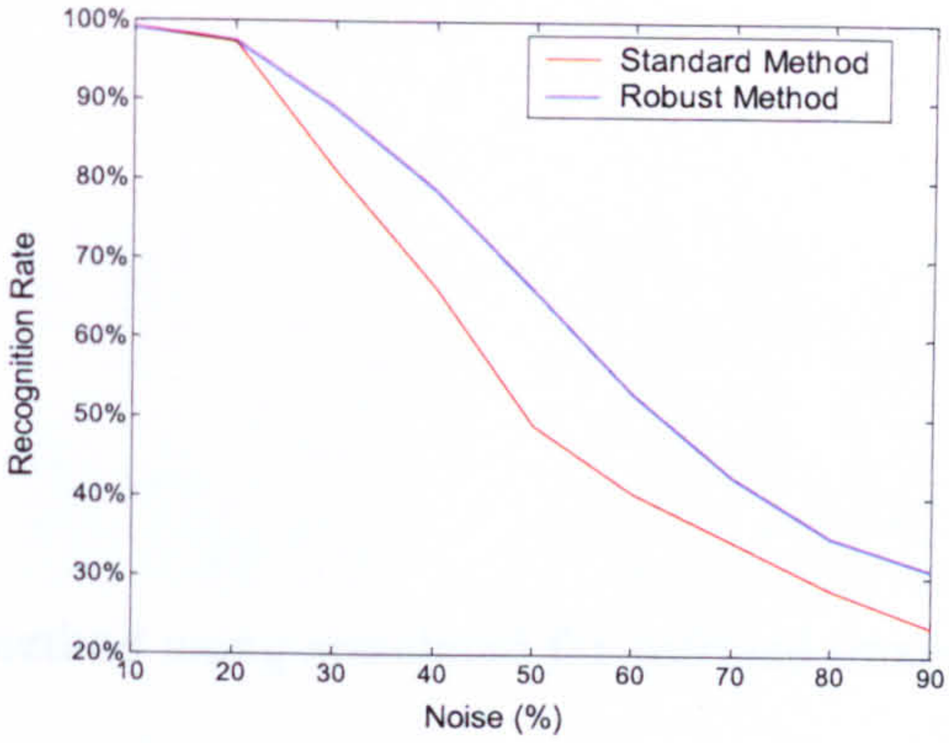
(a)



(b)



(c)



(d)

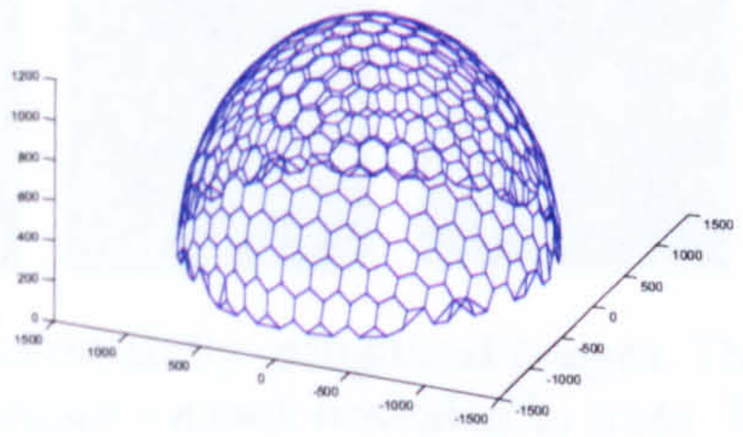
Figure 3-48 (a) The effects of three kinds of noise and occlusion at 30% region of the image; (b) comparison between two methods using images with black occlusion; (c) with white occlusion; (d) with salt and pepper noise

We examined images with white or black occlusion at noise levels up to 30% because certain objects, e.g. cars, are already completely occluded in some orientations.

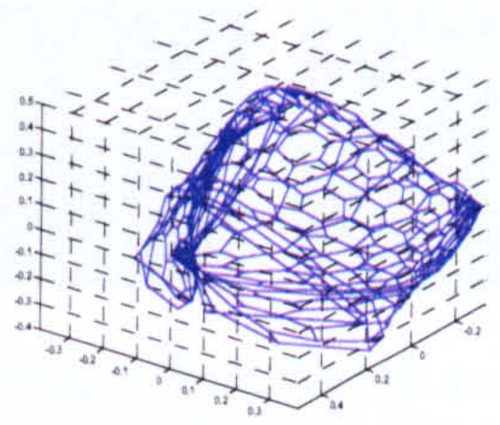
Result and Conclusion:

Figure 3-48 shows the recognition rate of different levels of noise with both the standard and robust methods. We see that under all three noise conditions, the robust method works better than the standard method. This advantage is more significant in random 'salt and pepper' images than images with local region occlusion. This is because in images with local region occlusion, the algorithm is more likely to reach a local optimization which is incorrect. For example, compared with random noise, the occlusion is more likely to generate patterns in the image that is more closely fitted to another object.

(a)



(b)



(c)

Figure 3-49 (a) Object and (b) viewpoints of the training set in experiment 3 (c) 3D Eigenspace of training set

Experiment 3

Objective:

In this experiment, we examine the robust method using simulated far-infrared images (8-12μm). The test images are simulated scene with the target vehicle, trees and other vehicles. The aim is to evaluate the algorithm on simulated real world.

Procedure and Results:

We use simulated infrared images of a tank (see Figure 3-49 a) from 624 viewpoints in an upper viewing sphere centred on the tank (see Figure 3-49 b) and used these images as our training set. Figure 3-49 (c) shows the first three eigenvectors only of the Eigenspace generated from the training set.

In the "recognition" stage, we used some cluttered scenes to do pose estimation using the robust method. We designed two kinds of cluttered scene: 1) Cluttered by trees and 2) Cluttered by trees and other vehicles. Figure 3-50 shows some examples of images in which the pose of the tank has been successfully recovered. This does not constitute recognition, in the sense that the tank is the only possible object, but recovery of the pose does give indication of the possible deployment of the method. For example, if we use the individual Eigenspace method, then we effectively add manifolds to Figure 3-49 (c) and

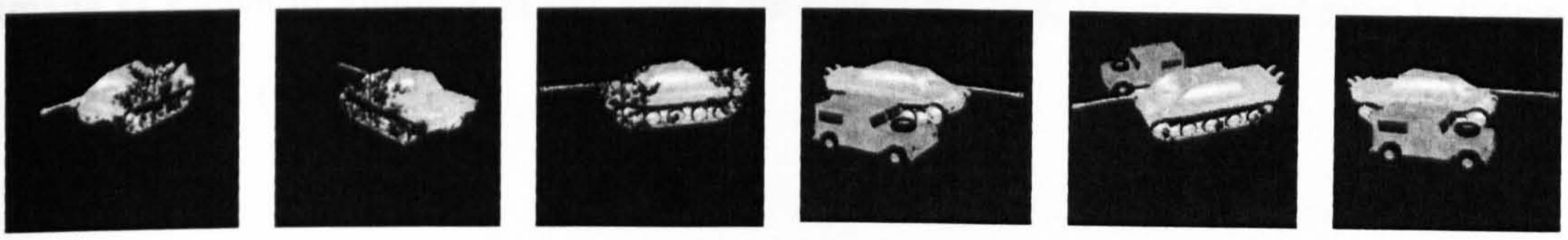


Figure 3-50 Examples of successfully recognized images. The 3 images from the left are of relatively simple cluttered scenes – a tank occluded by trees. The 3 images from the right are of relatively complex cluttered scenes –a tank occluded by trees and other vehicles

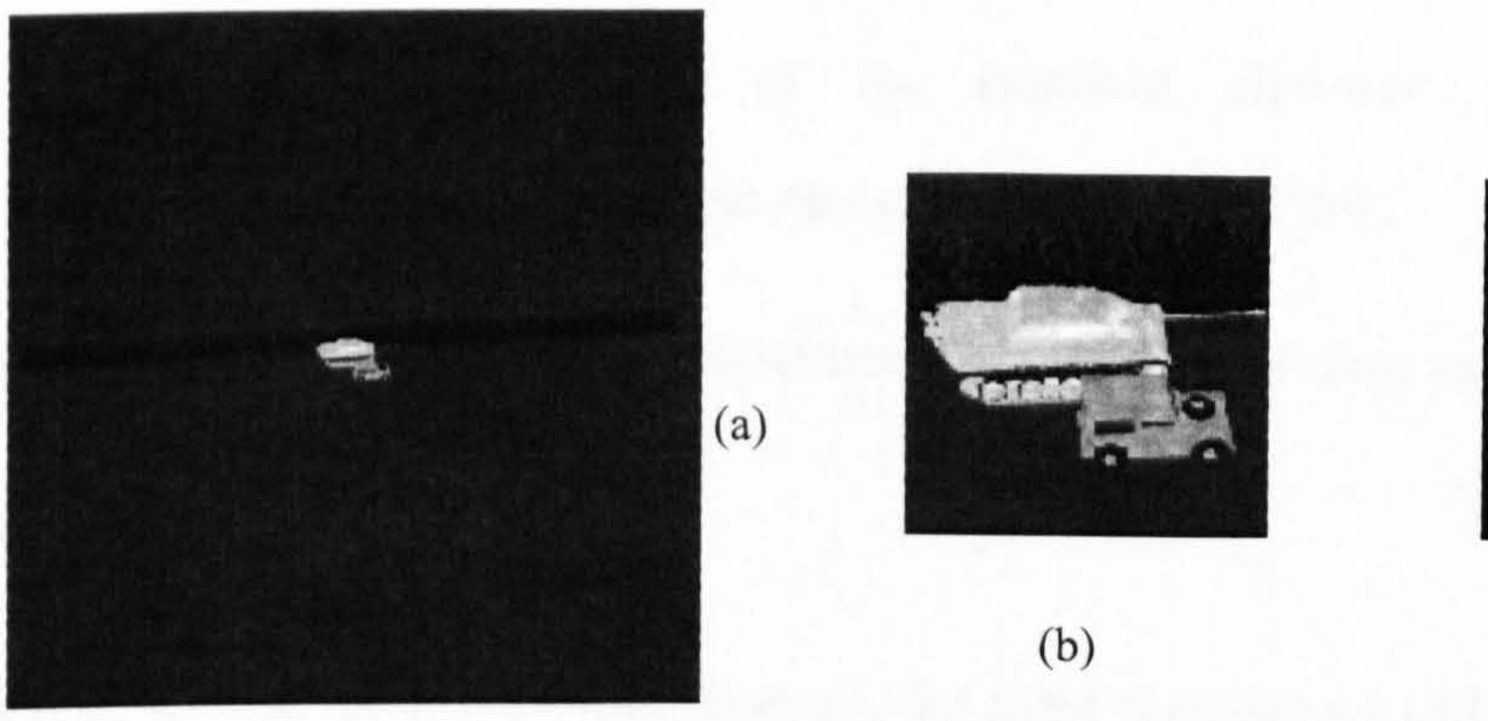


Figure 3-51 (a) One image frame in a video (b) Segmented image (c) recognized pose

the recognition problem becomes finding the distance to the nearest manifold to determine the object and pose simultaneously, as was the case in the earlier experiments.

The test images in Figure 3-50 are of the same size as the images in the training set. In these experiments, we are also interested in testing the method in a large scene where the object occupies only a very small part of the whole image. Figure 3-51 (a) shows one image from a video sequence. The only difference between the two images is the slightly changed position of the tank. One possible reason for using a video sequence is that difference images formed between adjacent frames can lead to motion-guided segmentation, i.e. to direct a sampling process to specific regions of interest in the same way that brighter (hotter) pixels can also be used in a selective process in a single image. Owing to the continuity between the two images, we compare them and crop the part within which the pixel brightness changes. We crop the changed part to form the final test

image as shown in Figure 3-51 (b). Using the robust method, the estimated pose is shown in Figure 3-51 (c).

3.6.6 Testing the Probabilistic Method in Dealing with small in-plane transformations

Objective:

(i) Evaluate the tolerance of the standard algorithm to small in-plane transformations, e.g., object centre moving up and left;

(ii) Testing the proposed Probabilistic adjusted method described in Section 3.5.4.

Procedure:

In this test, we use the leave-6-out strategy, the same training set and original test images as in Section 4.4.3. The difference is, in order to test the small in-plane transformation, we change the original test images to new test images by moving the centre of the object several pixels up and left. We use 100 eigenvectors in the recognition process.

First, we tested the tolerance of the standard algorithm to a small in-plane transformation. To test the effect of left or right movement, we made the test images by moving the object to the left by 8 distances: 2, 4, 6, 8, 10, 12, 14, 16 pixels in a 200*200 pixels image. Similarly, to test the effect of up or down movement, we made the test images by moving the object up for 8 levels: 2, 4, 6, 8, 10, 12, 14, 16 pixels in a 200*200 pixels image.

Second, we tested the probability based adjusted method and compare it with standard method. In the test, the test images were made by moving the object up to a random chosen level (out of the 8 levels in the previous test), so the recognition system doesn't have any prior knowledge about the small in-plane transformation.

Result and Conclusion:

Figure 3-52 show the results of small horizontal in-plane movement. We see that as the object centre moves left, the recognition rate gets worse generally. The following table shows the results of moving 10 pixels left. From the table, we see that the 3 easy confused objects, objects 5, 8 and 9 have a very low recognition rate. The recognition rate of other objects, e.g., object 6 and object 11, go down below 80% as we move the centre of the object 10 pixels to the left in a 200x200 pixels image.

Test image / Recognize as	obj1	obj2	obj3	obj4	obj5	obj6	obj7	obj8	obj9	obj10	obj11
obj1	**84**	0	0	0	0	1	0	0	0	0	0
obj2	0	**84**	0	0	0	0	0	0	0	0	0
obj3	0	0	**70**	12	2	0	0	0	3	0	5
obj4	0	0	10	**72**	9	3	0	2	3	0	3
obj5	0	0	1	5	**57**	1	0	45	60	3	12
obj6	0	0	0	2	1	**65**	2	0	1	23	3
obj7	0	0	0	0	0	0	**81**	0	0	5	0
obj8	0	0	0	0	4	0	0	**34**	10	0	0
obj9	0	0	3	1	14	0	0	7	**30**	1	7
obj10	0	0	0	0	3	16	1	0	1	**73**	0
obj11	0	0	0	2	0	1	0	0	1	0	**58**
Recognition Rate (%)	**100.**	**100.**	**83.3**	**85.7**	**67.9**	**77.4**	**96.4**	**40.5**	**35.7**	**86.9**	**69.0**
The object moved left at 10 pixels distance											

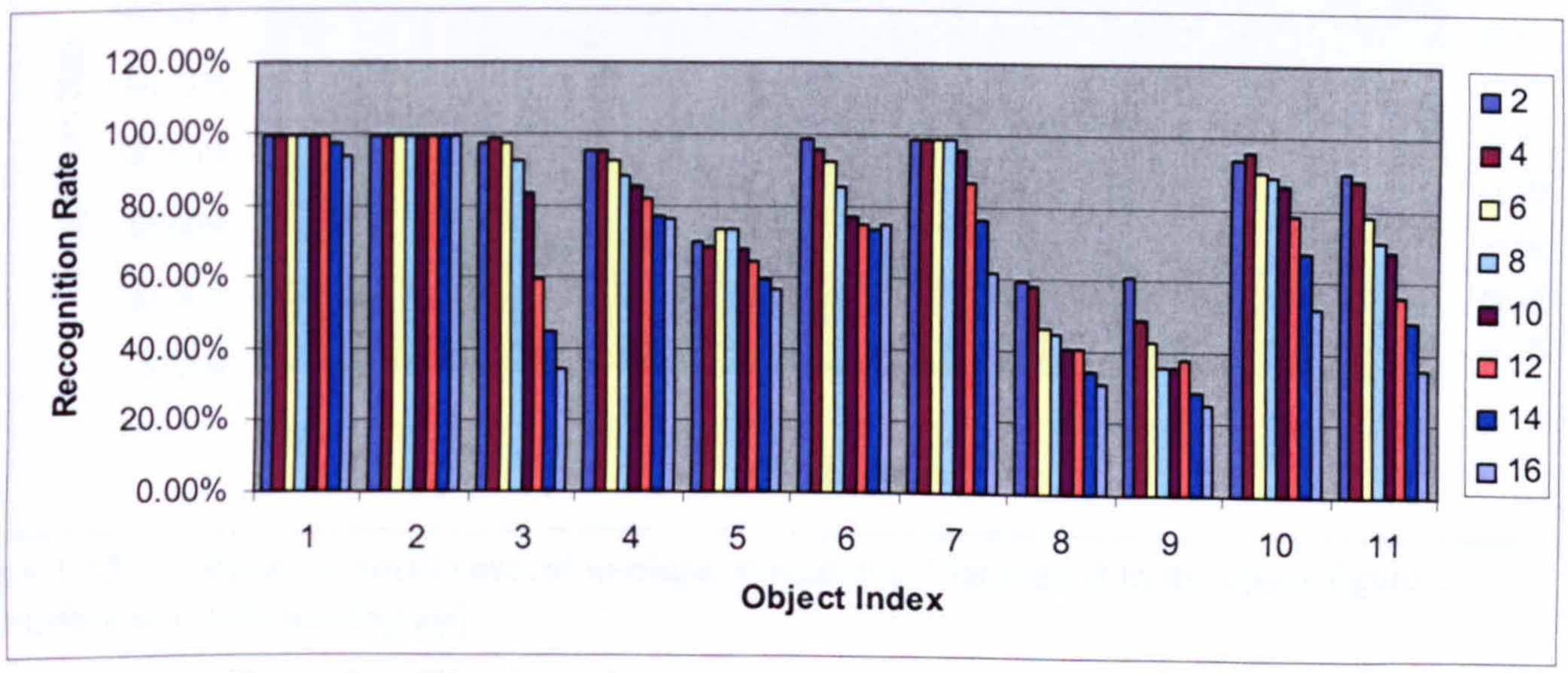


Figure 3-52 Effect of small horizontal in-plane movement (The legend in the upper figure is the number of pixels moving left)

Figure 3-53 shows the results of small vertical in-plane movement. The following table shows the results of moving 10 pixels up, the results of other levels are in Appendix A3. We see that not only are the recognition results of the three cars, objects 5, 8 and 9, getting worse, the recognition results of another object group – the three tank, objects 4, 6 and 10 are getting worse as well.

Test image / Recognize as	obj1	obj2	obj3	obj4	obj5	obj6	obj7	obj8	obj9	obj10	obj11
obj1	**60**	0	0	0	0	0	0	0	0	0	0
obj2	0	**84**	0	0	0	0	0	0	0	0	0
obj3	0	0	**51**	0	1	0	0	0	2	0	1
obj4	0	0	5	**36**	3	0	0	0	4	0	2
obj5	0	0	7	3	**45**	0	14	49	41	24	11
obj6	0	0	0	42	0	**14**	0	0	0	11	0
obj7	22	0	14	0	11	0	**66**	0	6	5	3
obj8	0	0	3	0	1	0	0	**0**	0	0	0
obj9	0	0	1	2	7	0	0	10	**8**	5	0
obj10	2	0	0	1	1	70	4	0	1	**36**	4
obj11	0	0	3	0	15	0	0	25	22	3	**63**
Recognition Rate (%)	**71.4**	**100.**	**60.7**	**42.9**	**53.6**	**16.7**	**78.6**	**0.0**	**9.5**	**42.9**	**75.0**
The object moved up at 10 pixels distance											

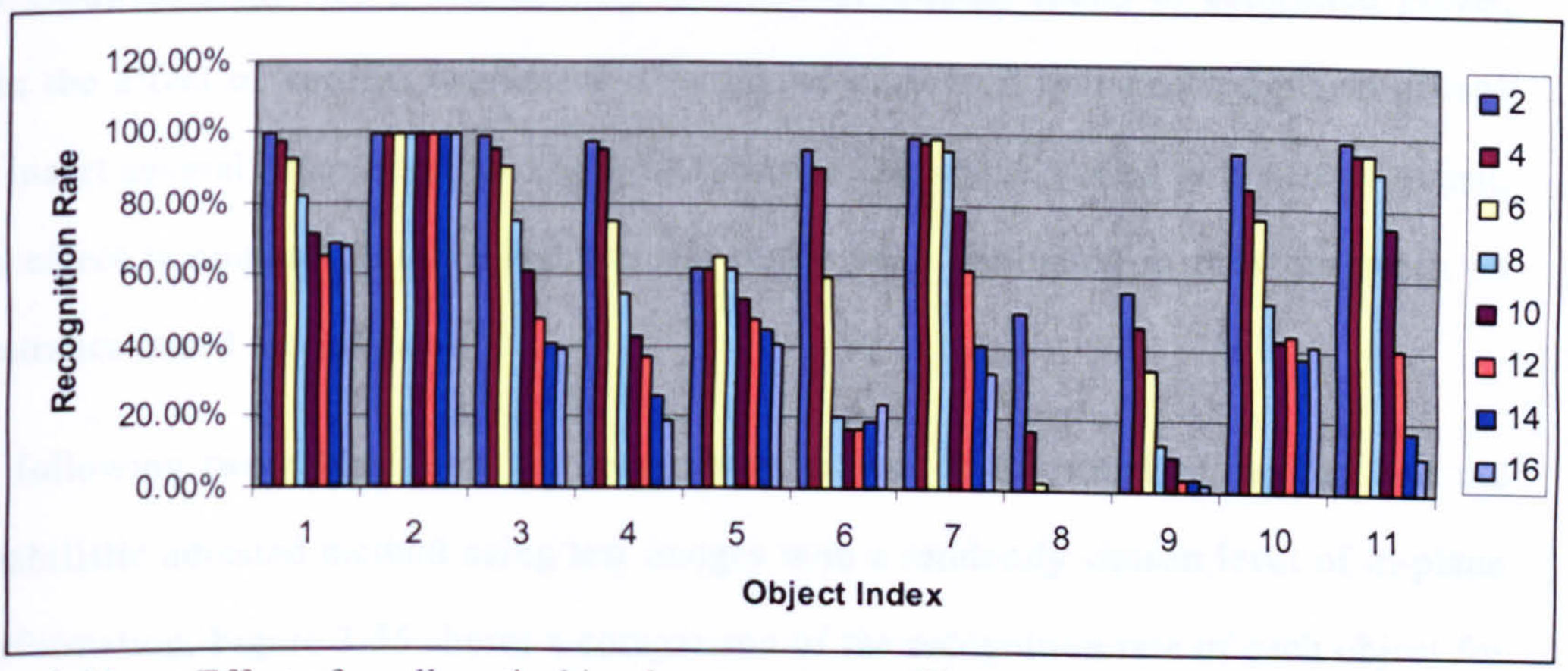


Figure 3-53 Effect of small vertical in-plane movement (The legend in the upper figure is the number of pixels moving up)

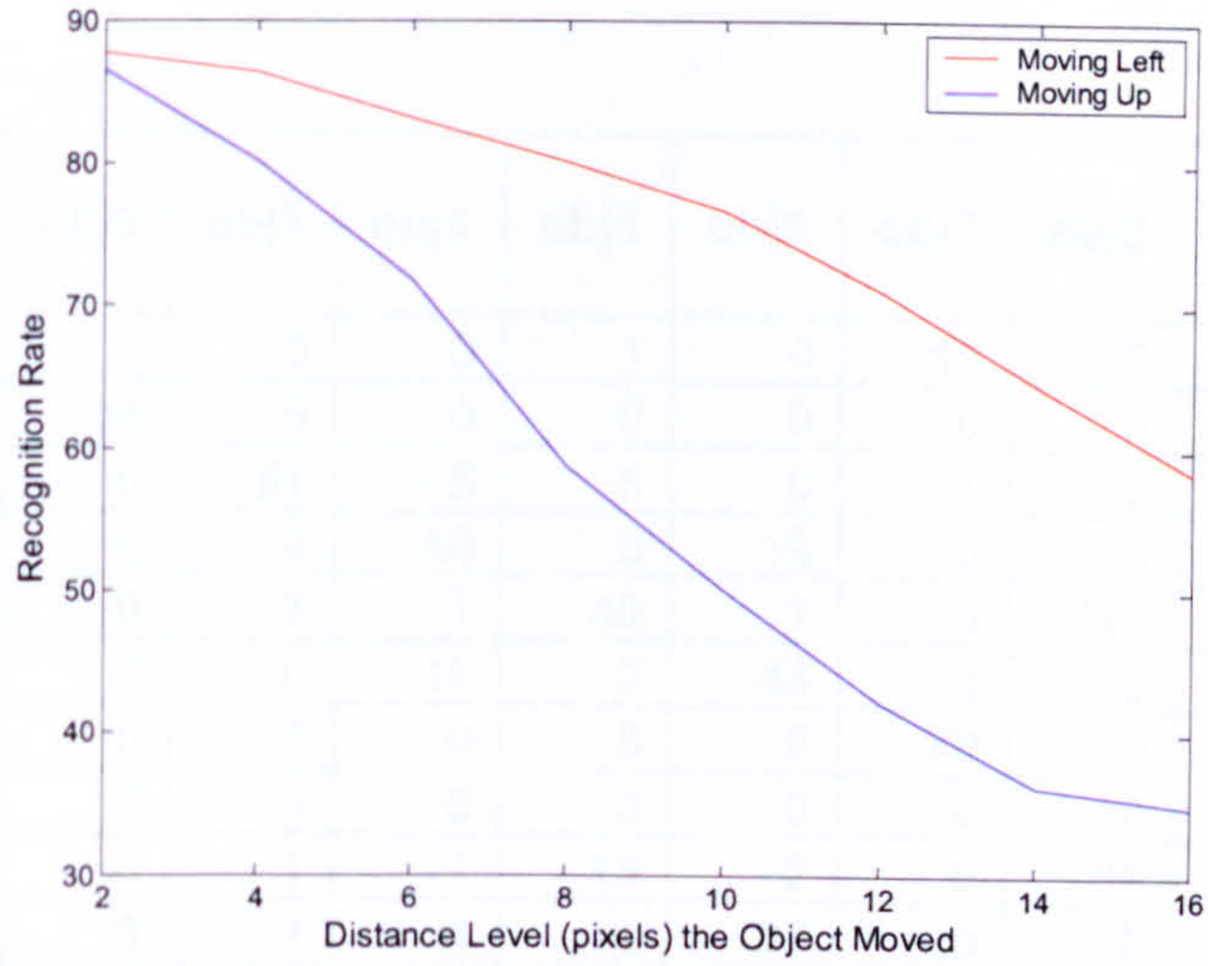


Figure 3-54 An comparison of the overall recognition rate when object centre moving up and left

Figure 3-54 shows a comparison between the effects of small horizontal movement and small vertical movement. We see that they both affect the performance of the algorithm in terms of recognition rate dramatically and the result of vertical movement is worse. This is because, in this algorithm, the test image is reshaped into a vector by reading the pixels up to down and then left to right. The effect of horizontal movement is equal to removal of a group of connected pixels and the insertion of another group of connected pixels, while the effect of vertical movement is to get rid of several unconnected pixels groups and insert several other pixel groups which change the image vector to a greater extent. This effect is counter intuitive and has not really been addressed in previous work on appearance based recognition.

The following two tables show the recognition matrix of the standard method and the probabilistic adjusted method using test images with a randomly chosen level of in-plane transformation. Figure 3-55 shows a comparison of the recognition rate of each object for both methods. We see that for any object, the probabilistic method improves the recognition performance dramatically.

Test image / Recognize as	obj1	obj2	obj3	obj4	obj5	obj6	obj7	obj8	obj9	obj10	obj11
obj1	**67**	0	0	0	1	0	17	0	0	0	0
obj2	0	**84**	0	0	0	0	0	0	0	0	0
obj3	0	0	**61**	5	8	0	7	3	2	1	0
obj4	0	0	4	**58**	3	19	0	0	4	4	3
obj5	1	0	7	1	**49**	1	16	42	42	12	16
obj6	0	0	0	19	0	**44**	0	0	2	42	0
obj7	16	0	7	0	8	0	**70**	0	5	10	2
obj8	0	0	3	0	3	0	0	**10**	29	0	14
obj9	0	0	1	1	13	2	0	21	**26**	4	9
obj10	0	0	1	4	1	37	5	0	3	**62**	4
obj11	0	0	0	0	7	0	0	14	9	0	**66**
Recognition Rate (%)	79.8	100.	72.6	69.0	58.3	52.4	83.3	11.9	31.0	73.8	78.6

The Standard Method

Test image / Recognize as	obj1	obj2	obj3	obj4	obj5	obj6	obj7	obj8	obj9	obj10	obj11
obj1	**82**	0	0	0	0	0	2	0	0	0	0
obj2	0	**84**	0	0	0	0	0	0	0	0	0
obj3	0	0	**83**	2	1	0	0	0	0	0	0
obj4	0	0	1	**74**	2	7	0	0	2	0	0
obj5	0	0	0	1	**54**	1	0	17	34	0	1
obj6	0	0	0	7	0	**72**	0	0	0	11	0
obj7	2	0	0	0	0	0	**83**	0	0	5	0
obj8	0	0	0	0	9	0	0	**42**	53	0	0
obj9	0	0	0	1	18	0	0	34	**47**	0	1
obj10	0	0	0	0	0	11	1	0	0	**79**	3
obj11	0	0	0	0	1	0	0	0	1	1	**82**
Recognition Rate (%)	97.6	100.	98.8	88.1	64.3	85.7	98.8	50.0	56.0	94.0	97.6

Probabilistic adjusted method

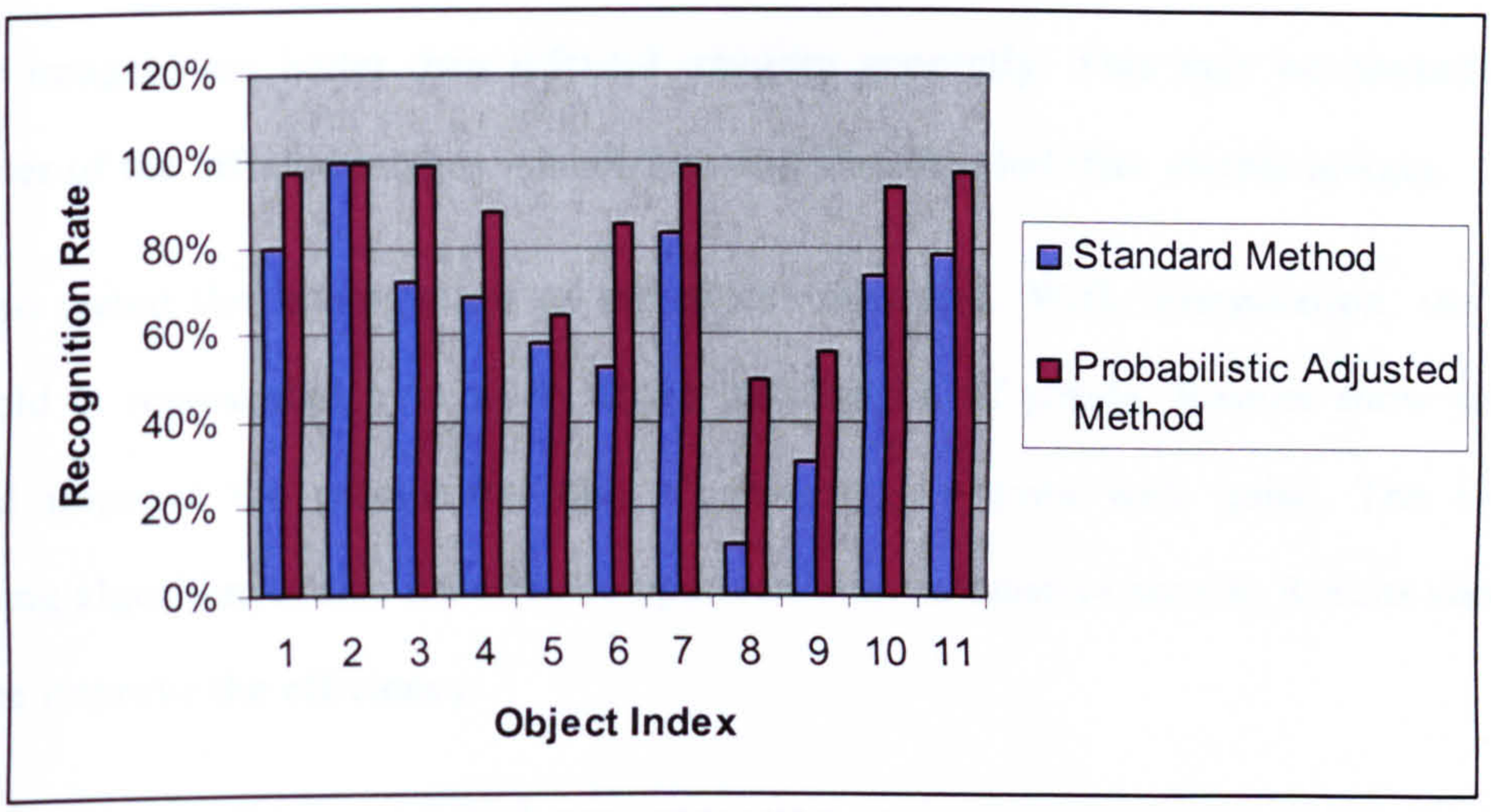


Figure 3-55 Comparison between results of original method and probabilistic method

3.6.7 Conclusion

In this section, we have tested the appearance based object recognition method on both visible and infrared imagery. When testing with visible imagery, we used the Coil-100 image set with 100 objects and 36 views of each as training images and 36 other views as test images. We tested using different number of training objects and various number of eigenvectors. Results show that the less the number of training objects and the more the number of eigenvectors the higher the recognition rate. Generally, 10 eigenvectors are enough for object recognition in a less than 20-objects Eigenspace. When testing in infrared imagery, we use a database of 11 vehicles generated by CameoSim package. In the database, each object is represented by 337 images of different views – thermal state differences are not considered at this stage and will be discussed in Chapter 5. We use the leave-six out strategy, i.e., using randomly chose 331 images as training images and the maintaining 6 as test images. There are three objects in the database that appear similar in infrared imagery. The average recognition rate for 11 objects is between 80%-90% using 10-100 eigenvectors. If we get rid of the effect of these three objects, the recognition rate tends to be stable at above 96% using 20 eigenvectors. If we compare the performance of the appearance based algorithm in visible and infrared imagery, although there are difference in the objects used, viewpoint selected and procedure used, the results in

visible imagery are better than infrared imagery generally. This may be caused by the character of the infrared images – more blur and less detailed than visible images.

We also tested the interpolation of the object manifold. With interpolation, the object manifold is represented by a more dense distribution of points. Results show that this method improve the recognition rate of new images even with noise. The k-d tree searching algorithm is also tested in comparison with exhaustive search. Results show that k-d tree improve the efficiency.

Bad imaging conditions and bad pre-processing, e.g., segmentation, of the image can cause lower resolution test images or test images with a small rotation in image plane. Tests show that these do not affect the result of appearance based recognition very much.

In the experiments, we have tested the performance of the robust sampling method in recognizing noisy images using different measurements, e.g., projection distance and recognition rate. The first measurement, projection distance, is the distance between the original image and noisy image in Eigenspace. For any appearance-based method aiming at minimize the effect of noise, we would expect it to minimize the projection distance. The ideal case is that the projection distance is zero, which means that the projection of the original image and of the noisy image are identical and the method totally eliminates the effects of noise. Compared to the standard method, the random sampling method improves the result considerably, e.g., the projection distance is low and stable when up to 50% area of the image is noisy. Using recognition rate as a measurement, we tested black occlusion, white occlusion and random noise. For all three kinds of noise and occlusion, the recognition rate of the random sampling method is higher than standard method. We also tested the method using infrared images with more complex noise, e.g., trees in the background, trees in front of the object, or the object occluded by other objects, and the method could successfully identify the object in these scenes.

For small in-plane transformations, we have tested the object moving left and up in the image plane by 8 different distances. The 8 distances are 2 pixels to 16 pixels, at 2 pixel

intervals. Using the standard method, the results show that the recognition performance degrades as the object is moved by more pixels. To test the performance of the proposed image window method based on the probabilistic framework (see section 3.5.4), we generated test images which transformed in the image plane by random distance. In the recognition stage, we don't have any prior knowledge about the level of in-plane transformation. The proposed method could improve the overall recognition rate from 65% to 85%.

Chapter 4

Non-linear Embedding Methods

In standard appearance-based recognition algorithms, we use Principal Component Analysis (PCA) as the basic technique in feature extraction. PCA, as a linear dimensionality reduction method, is used to

- Rotate the original feature space and find a projection direction that has maximum variance
- Reduce dimensionality by projecting high-dimensional data onto a low-dimensional subspace

In other words, PCA identifies significant coordinates and linear correlations in the original, high-dimensional data. For example, we have 20 images of 128x128 pixels. Originally, each image is represented as a point in a 128x128 dimensional space. Using PCA, we can embed each image in the image set with a 20 dimensional space, the Eigenspace (see Figure 3-6). Each dimension in Eigenspace is a linear rotation from the original dimensions. This 20-dimensional Eigenspace is enough to discriminate all the images in the image set and reconstruct the images to a certain accuracy.

Although using PCA we achieve the dimensionality reduction, the basis of PCA does not reflect the perceptually meaningful structure of the images. In this Chapter, we consider other nonlinear dimensionality reduction methods which are claimed to extract

meaningful dimensions and discover nonlinear structure embedded in the image set. We then examine one of them, Isomap, in the context of object recognition.

4.1 A review of Non-linear Dimensionality Reduction Methods

For linearly embedded manifolds, PCA is guaranteed to discover the dimensionality of the manifold and produce a compact representation in the form of an orthonormal basis. However, PCA is completely insensitive to higher-order, nonlinear structure. The fact that PCA is a linear method implies a potential oversimplification of the datasets being analysed. It is therefore appropriate when a simple, linear, globally applicable rule for extracting information from new data points exists. It is unsuitable when the correlations are nonlinear or when no such simple rule exists [121] [122].

To overcome this limitation of PCA, many extensions have been suggested. Principal curves [123] are smooth one-dimensional curves that pass through the middle of the data points, providing a nonlinear summary. PCA is regarded as the special case of a straight line. Principal curves can be extended to several dimensions and called principal surfaces. However, the method is not suitable for very high-dimensional problems [124]. Another extension is local PCA [125], which clusters the data points and performs PCA within each cluster. However, it does not map the data into a simple global lower-dimensional coordinate system.

Two important nonlinear methods are Isometric Feature Mapping (Isomap) [126][127] and Locally Linear Embedding (LLE) [128] [129]. LLE computes a different local quantity. It calculate the best coefficients to approximate each point by a weighted linear combination of its neighbours, and then tries to find a set of low-dimensional points, which can be linearly approximated by its neighbours with the same coefficients that were determined from the high-dimensional points. Isomap extends MDS by a sophisticated distance measurement to achieve nonlinear embedding. They build a graph using data that is only locally connected, and then measure pairwise distances by the length of the shortest path

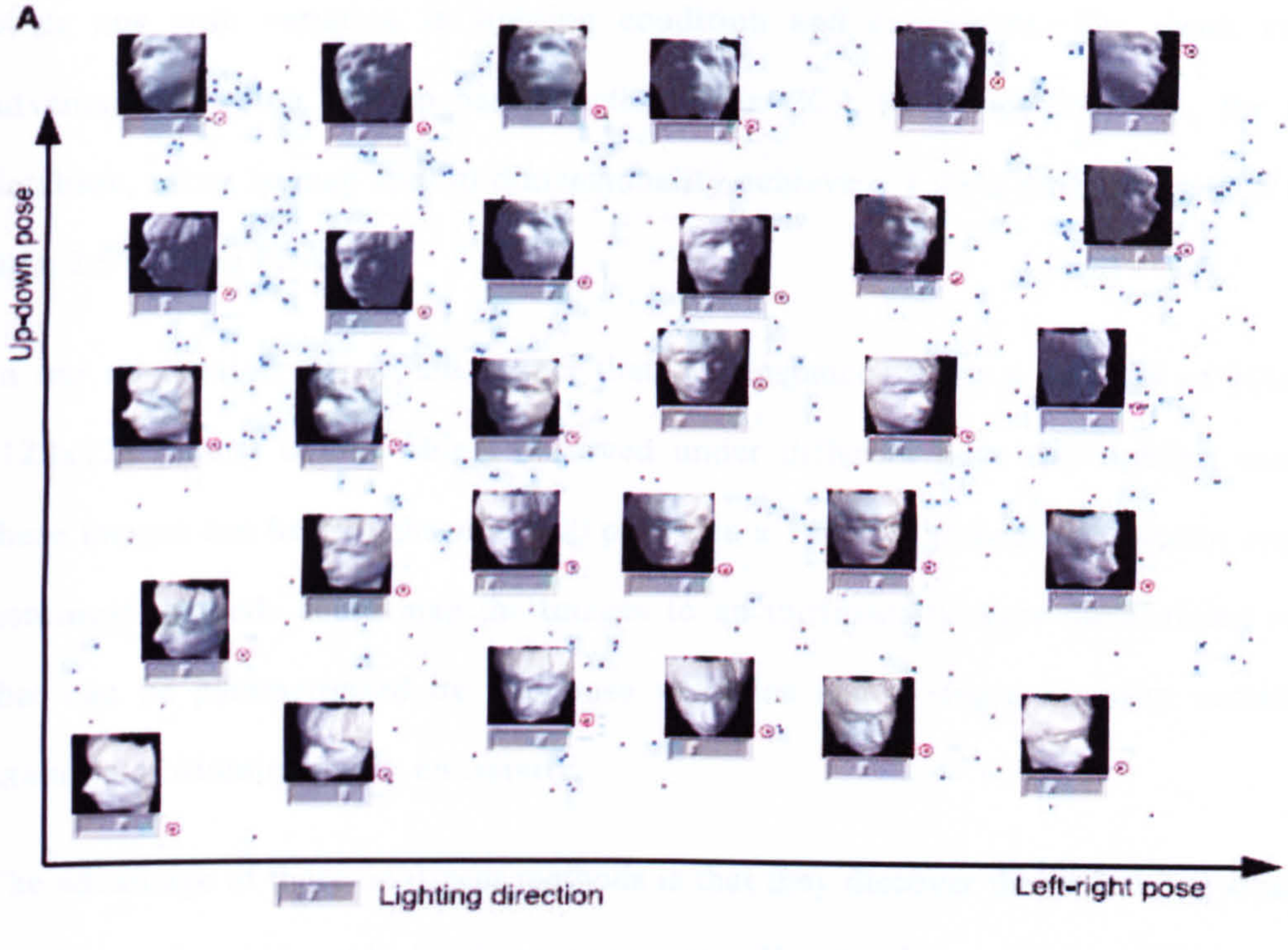


Figure 4-1 Isomap reduce the high dimensional image vectors into a 3-dimensional space. Source from [126]

on that graph. Finally, MDS is used to find a set of low-dimensional points with similar pairwise distances.

Using Isomap, Tenenbaum etc. [126] embedded a sequence of 4096-dimensional vectors, representing the brightness values of 64x64 pixel images of a face rendered with different poses and lighting directions, within a 3-dimensional space (see Figure 4-1). Although the input dimensionality is quite high, the perceptually meaningful structure of these images has many fewer independent degrees of freedom. In the new feature space, all of the images lie on an intrinsically 3-dimensional manifold that can be parameterized by two pose variables plus an azimuthally lighting angle. The goal of Isomap is to discover a low-dimensional representation with coordinates that capture the intrinsic degrees of freedom of a data set.

Yang [130] proposed a method that extends Isomap with the Fisher Linear Discriminant and created a so called extended Isomap for pattern classification. They examined the proposed method with two face databases, one with variation in pose and scale and the

other one with variation in lighting condition and expression. The result shows an advantage of using Isomap based method than PCA based method, e.g., for the first database, using Isomap and 30 dimensionality achieve a 1.75% error rate which is better than 2.5% using PCA.

In our application, we would expect that, for instance, given a data set of 100 images (128x128 pixels) of one object observed under different pose and lighting conditions, these images can be thought of as 100 points in a 128x128 dimensional vector space. The nonlinear methods could map the images to an intrinsically three-dimensional manifold that can be parameterised by two pose variables and a single intensity variation, for example of illumination or emissivity.

The advantage of these nonlinear methods is that they discover the underlying structure of the dataset by using only local measurements. The disadvantage caused by using these local measurements is that it cannot find a global application rule for extracting information from new data points. If applied in appearance-based recognition, the nonlinear methods can improve the training process but make the recognition process much more complex. Another disadvantage is that these approaches are limited to embedding within a single manifold. If we introduce multiple manifolds to such approaches, they tend to capture the intrinsic structure of each manifold separately without generalizing to capture inter-manifold aspects.

In the following sections, we will describe Isomap methods and perform experiments using our own image data sets. We then discuss the application of these methods in appearance based recognition.

4.2 Isomap as a dimensionality reduction method

Isomap is another dimensionality reduction method initially proposed by Tenenbaum [126] [127] to find meaningful low-dimensional structure hidden in high-dimensional observations. We can compare Isomap to the basic PCA approach.

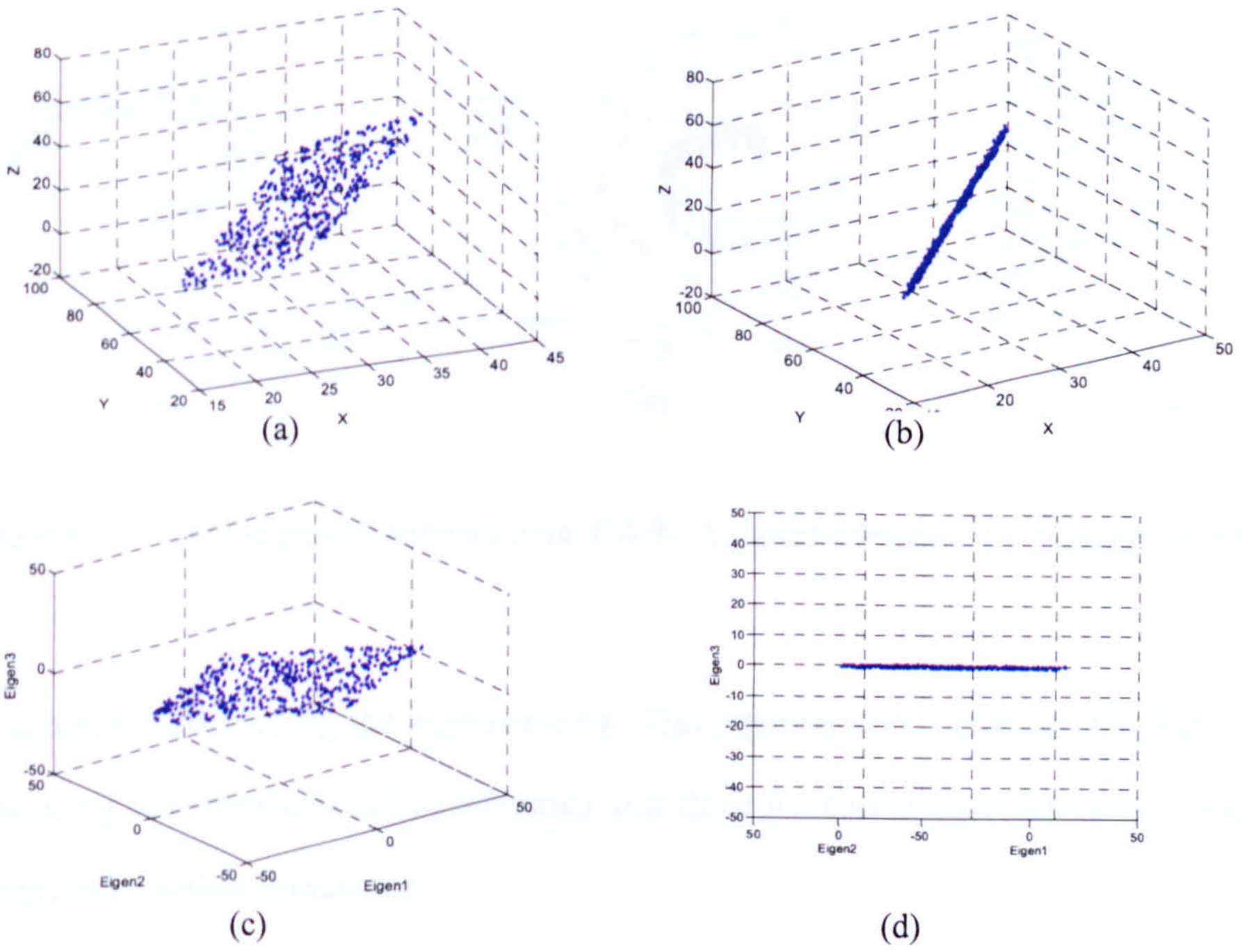


Figure 4-2 (a)(b) Two views of a plane embedded in a 3D space (c)(d) Two views of the PCA based subspace which discover the true dimensionality of the plane shown in (a)(b)

4.2.1 PCA, MDS and Isomap

Isomap can be thought of as an extended MDS (Multidimensional Scaling) [131] by using a sophisticated distance measurement for the data. In this section, we compare the PCA, MDS and Isomap. PCA (Principal Component Analysis) [132] and MDS are two widely used dimensionality reduction methods to discover the true structure of data lying on a linear subspace of the high-dimensional input space. Geometrically, PCA rotates the original space and find an orthonormal coordinate system so that the correlation between different axes is minimized and the primary axes of the data lie along the axes of the coordinate space.

Figure 4-2 shows an example of PCA (a)(b) a flat two-dimensional manifold has been linearly embedded in a three dimensional observation space; (c)(d) PCA discovers that the intrinsic dimensionality of the manifold is 2 and produces a compact representation in the form of a 2D orthonormal basis. This is done by calculating the covariance matrix of the

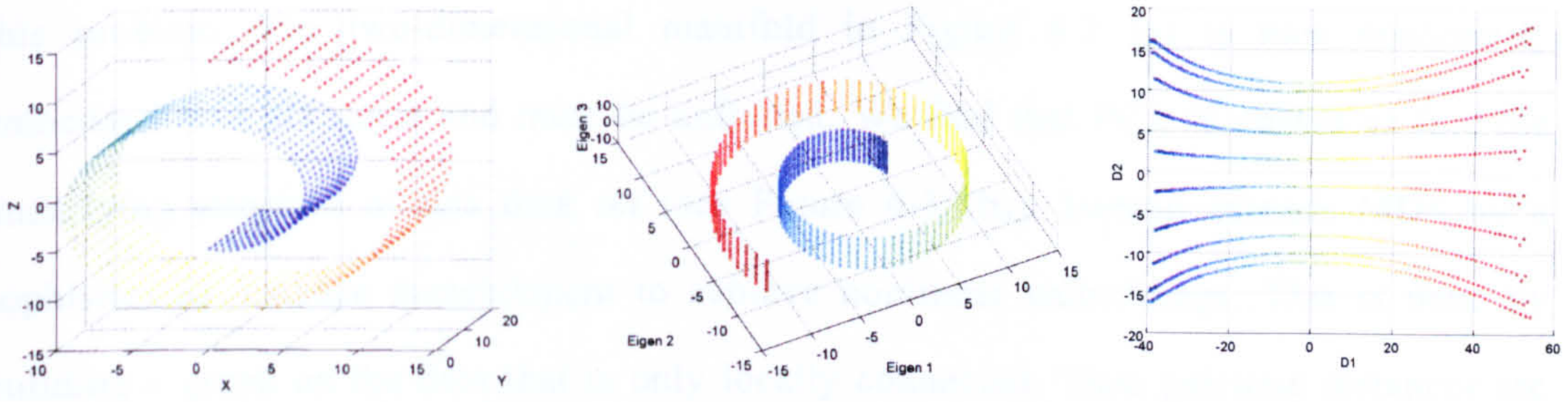


Figure 4-3 (a) Original Swissroll data; (b) PCA based subspace; (c) Isomap based subspace

input samples and finding the eigenvectors. The eigenvectors represent the dimensions of the resulting low-dimensional coordinates and the corresponding eigenvalues describe the total variance in that dimension.

MDS aims to represent the data points in a lower dimensional space while preserving as many of the pairwise similarities between the data points as possible, i.e., it plots similar objects close and dissimilar objects apart. In the simple case of MDS, metric MDS, these pairwise similarities are proportional to the distances of the corresponding points in the multidimensional space. The Minkowski distance metric provides a general way to specify distance in a multidimensional space:

$$d_{ij} = \left[\sum_{k=1}^{m} \left| x_{ik} - x_{jk} \right|^{r}\right]^{1/r}, \tag{4-1}$$

where m is the number of dimensions, and x_{ik} is the value of dimension k for stimulus i. With $r = 2$, the metric equals the Euclidian distance metric. Thus, the input to MDS is a square symmetric matrix indicating the relationships among a set of objects. Mathematically, metric MDS is performed by calculating the top eigenvectors of the distance matrix. Details of how MDS is implemented can be found in the third step of the implementation of Isomap (see next section). When the distances in MDS are Euclidian distances, the resulting subspace of MDS is the same as PCA.

Both PCA and MDS discover linear embeddings very well, however, in many applications data usually incorporates non-linear structures. Figure 4-3 shows a simplified version of

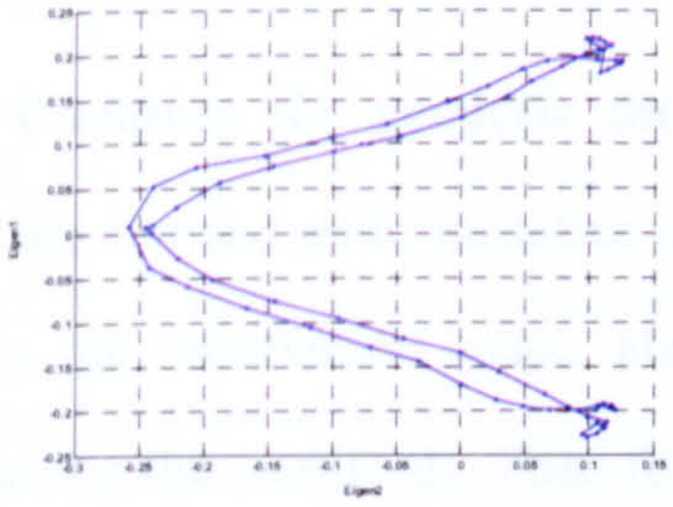
this problem. The two-dimensional manifold in Figure 4-3 (a) is now nonlinearly embedded in a 3D space and must be unfolded. We find that PCA is unable to find the underlying structure of this data set (see Figure 4-3 (b)). Isomap extends MDS by a sophisticated distance measurement to achieve nonlinear embeddings. This is done by building a graph on the data that is only locally connected. Then pairwise distances are measured by the length of the shortest path on that graph. This length is an approximation to the distance between its end points. The crux is estimating the geodesic distance between faraway points, given only input-space distances. In Figure 4-3 (c), we see that using Isomap, the 3 dimensional Swissroll unfolds to a plane.

4.2.2 Implementation and Characteristics of Isomap

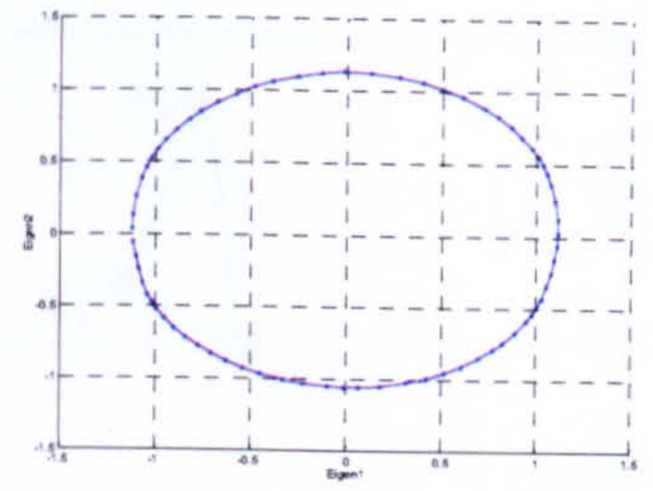
The implementation of Isomap can be detailed in 3 steps: building the neighbourhood map, computing the geodesic distances, and low dimensional embedding. In this section, we describe the details of these steps.

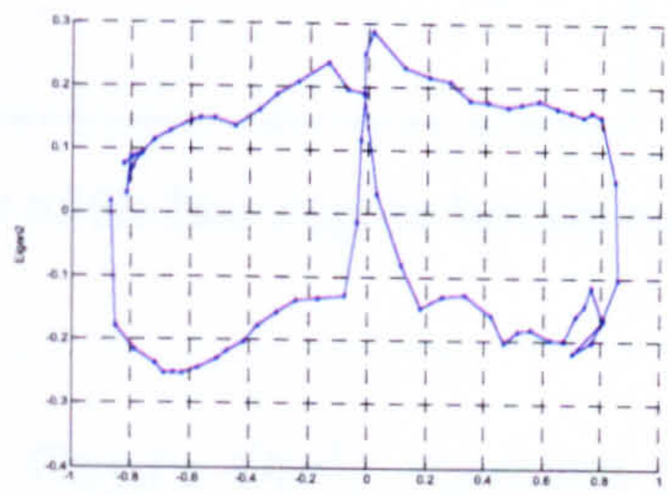
Step1 - Build Neighbourhood Map. The input is a matrix D that records all the pairwise distances between samples, e.g., if we have 8 samples, D is an 8 by 8 matrix. The neighbourhood of a point may be either the k nearest points or the set of point within a radius ε. A graph is then built by linking all neighbouring points and labeling all arcs with the Euclidean distance between the corresponding linked points. By the end of this step, D becomes D', in which the pairwise distances between two neighbours remain the same as in D, while the pairwise distances between two non-neighbours are infinity. This step makes each point connected with its neighbours, either the k nearest points or the set of points within a radius ε, and disconnected with all other points. For example, if we set $k=2$, only the distance between each high-dimensional point and its 2 neighbours are recorded and the distance between that point and all other points are infinity.

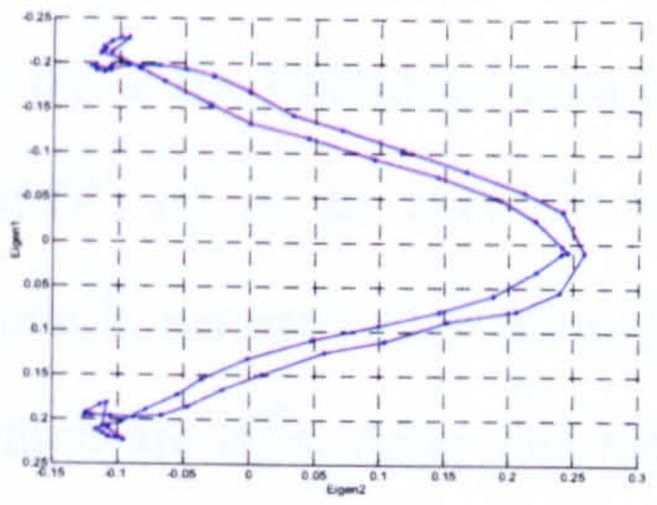
It is not known how to find the optimal parameter k or ε. However, the scale-invariant parameter k is typically easier to set than the neighbourhood radius ε. Tenenbaum pointed out that when the local dimensionality varies across the data set, the k nearest



(a)







(b)

Figure 4-4 (a) upper: an image of the object being modelled, lower: PCA based subspace representation of 72 continuously posed images of the object; (b) from top to bottom, Isomap based representation with 2, 8, and 70 local neighbours;

neighbour method may yield misleading results. In our application, we choose the k nearest neighbour method because our training images are uniformly sampled in the observation space.

In the resulting subspace, the position of each image is mainly decided by its neighbours, which are the similar images in the image set. The number of local neighbours is an important parameter for those local linear or called nonlinear methods. In Figure 4-4, we see that the number of neighbours can affect the subspace manifold dramatically. If the number is close to the total number of images in the image set, the manifold generated by nonlinear method can be identical to the PCA based manifold. This is because when we specify all other images as neighbours of each image, the local method becomes global.

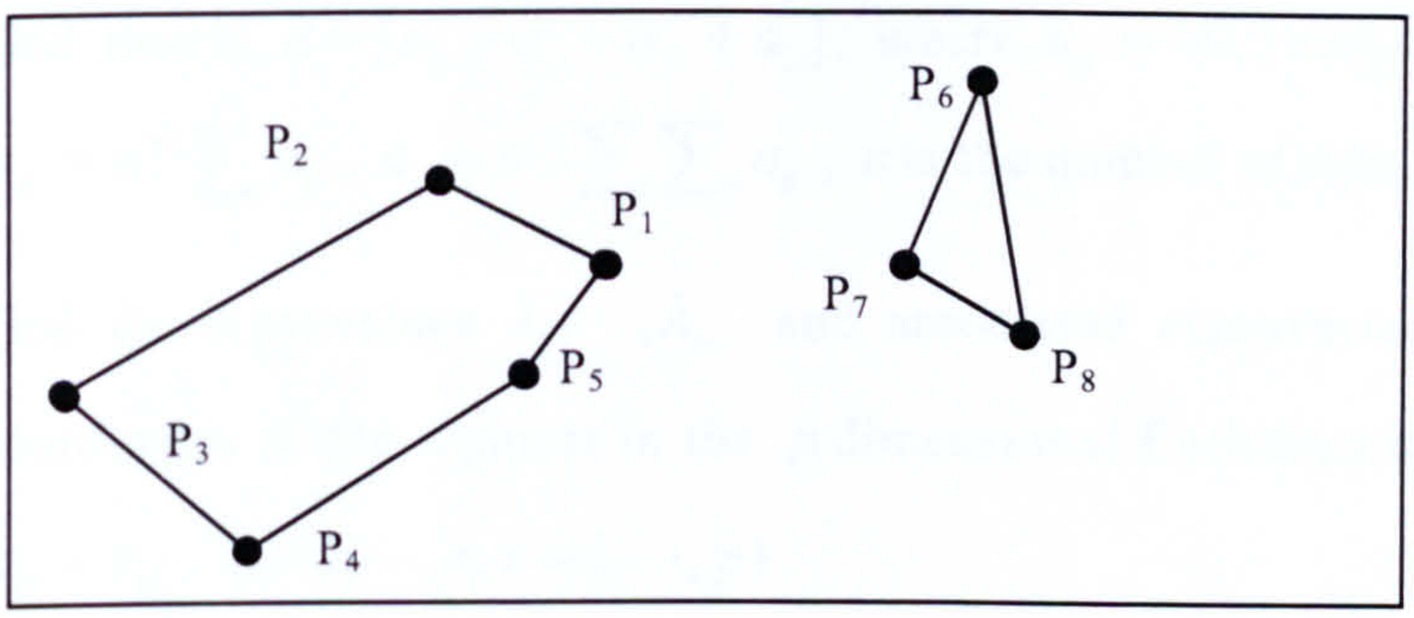


Figure 4-5 Illustration of the result of the first steps in Isomap method

Step2 - Compute the geodesic distances. Floyd's $O(r^3)$ algorithm [133] is used to compute the shortest path: Initialize $d_G^{ij} = d_E^{ij}$ (d_E^{ij} is the Euclidean distance between node i and j in the observation space, elements of D', d_G^{ij} is the geodesic distance between i and j along the manifold). Then for each node k, set each $d_G^{ij} = \min(d_G^{ij}, d_G^{ik} + d_G^{kj})$. By this step, D' becomes D_G, the elements of which are d_G^{ij}s. After this step, each point is not only connected with its k nearest neighbours, but also connected with its neighbours' neighbours.

Now we demonstrate the result of step 1 and 2 using a simple example, eight points in a 2-dimensional space (see Figure 4-5). Here we specify k =2 and look at the distance between P_1 and other points. After the first step, in the distance matrix, the distance between P_1 and P_2 is P_1P_2; the distance between P_1 and P_5 is P_1P_5; and the distance between P_1 and other points are infinity. After the second step, the point P_1 is connected with more points, e.g., the distance between P_1 and P_4 becomes $P_1P_5 + P_5P_4$ from infinity. However, after this step, each point is not guaranteed to be connected with all other points in the space. In this example, with $k = 2$, point P_1 is not connected with point P_6, P_7 and P_8. If we set $k = 3$, P_1 is connected with P_7 after the first step and all the eight points are connected after the second step.

Step3 - Low dimensional Embedding. In this step, classical MDS is applied to the approximated geodesic distance matrix D_G:

(i) Find matrix $B=[a_{ij}-a_{i.}-a_{j.}+a_{..}]$, where $a_{ij}=-0.5\times d_G^{ij\,2}$, $a_{i.}=n^{-1}\sum_j a_{ij}$, $a_{j.}=n^{-1}\sum_i a_{ij}$, $a_{..}=n^{-2}\sum_i\sum_j a_{ij}$, n is the number of data points;

(ii) Find the eigenvalues $\lambda_1,\cdots,\lambda_{n-1}$ and associated eigenvectors $\vec{v}_1,\cdots,\vec{v}_{n-1}$, the coordinates of the n points in the p dimensional Euclidean space are given by $x_{ir}=v_{ri}\quad(i=1,\cdots,n;\ r=1,\cdots,p)$.

4.3 Comparing Isomap and PCA for Recognition

In last section, we have shown that Isomap can find the true data structure of some non-linear data sets, e.g., the synthetic Swiss roll example, while PCA cannot. In this section, we compare the two methods in pose estimation and object recognition.

4.3.1 PCA vs. Isomap in Pose estimation

In pose estimation, one difficulty is to separate two poses which are taken from quite different viewpoints, yet look similar. For example, we consider the image set of a car in Coil20 with 72 different poses in Figure 4-6 (shows only 12 poses from 72 poses). To capture poses 1 to 12, the camera moves in a circle around the car (ignore the scale difference).

Figure 4-7(a) shows an Eigenspace (or PCA) representation of the image set. We see that for many poses, they are not only close to their continuing poses, but also close to some other poses. For example, pose 1 and pose 7 are not continuous in azimuth and elevation and in the images, the cars are facing different directions. But their Eigenspace projections are close to each other. This is because that this method is only based on the appearance of the object. Like many other object of interest, the car is exhibited symmetrically on one or more axes.

However, if we use Isomap to embed those images, these two similar images from different poses can be well separated. Figure 4-7 (b) shows the Subspace generated by

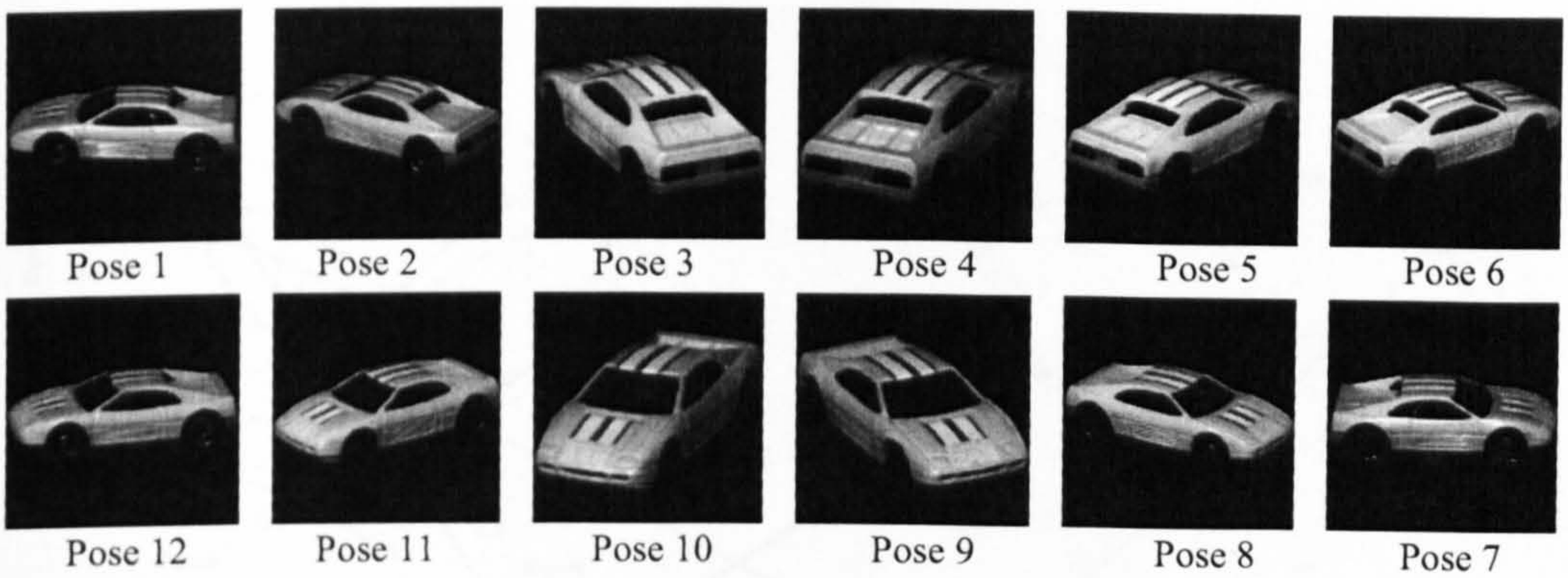


Figure 4-6 Images captured by moving the camera around the object, upper line: first half circle, lower line, second half circle

Isomap. We see that the manifold in Figure 4-7 (a) is unfolded and Pose 1 and Pose 7 are separated.

For both the PCA and Isomap subspace methods, the position in subspace of any single image depends on the appearance of the image itself and its neighbourhood. Using PCA, a global method, since pose 1 and pose 7 share the same neighbourhood and have similar appearance, their positions in subspace are close. Using Isomap, a local method, although poses 1 and 7 look similar, they have totally different neighbourhoods. For example, if we consider two local neighbours, the neighbours of pose 1 are poses 2 and 12 while the neighbours of pose 7 are poses 6 and 8. The different local neighbourhoods of pose 1 and 7 in the Isomap method make their projections in subspace far apart.

We have also compared the PCA and Isomap methods in pose estimation using a CameoSim image set. In the image set, each object is represented by 337 poses captured at the vertex positions of an upper sphere of a third-level Icosahedron (see Figure 3-38(b)). In the observation space, each pose has 6 nearest poses which are the images taken in the 6 nearest camera positions. Here we use images of object 2, landrover (see Figure 3-38 (a)), as the testing object. To do pose estimation, we use the leave-one-out strategy. If the pose is identified as one of its 6 nearest poses, we say it is correctly identified.

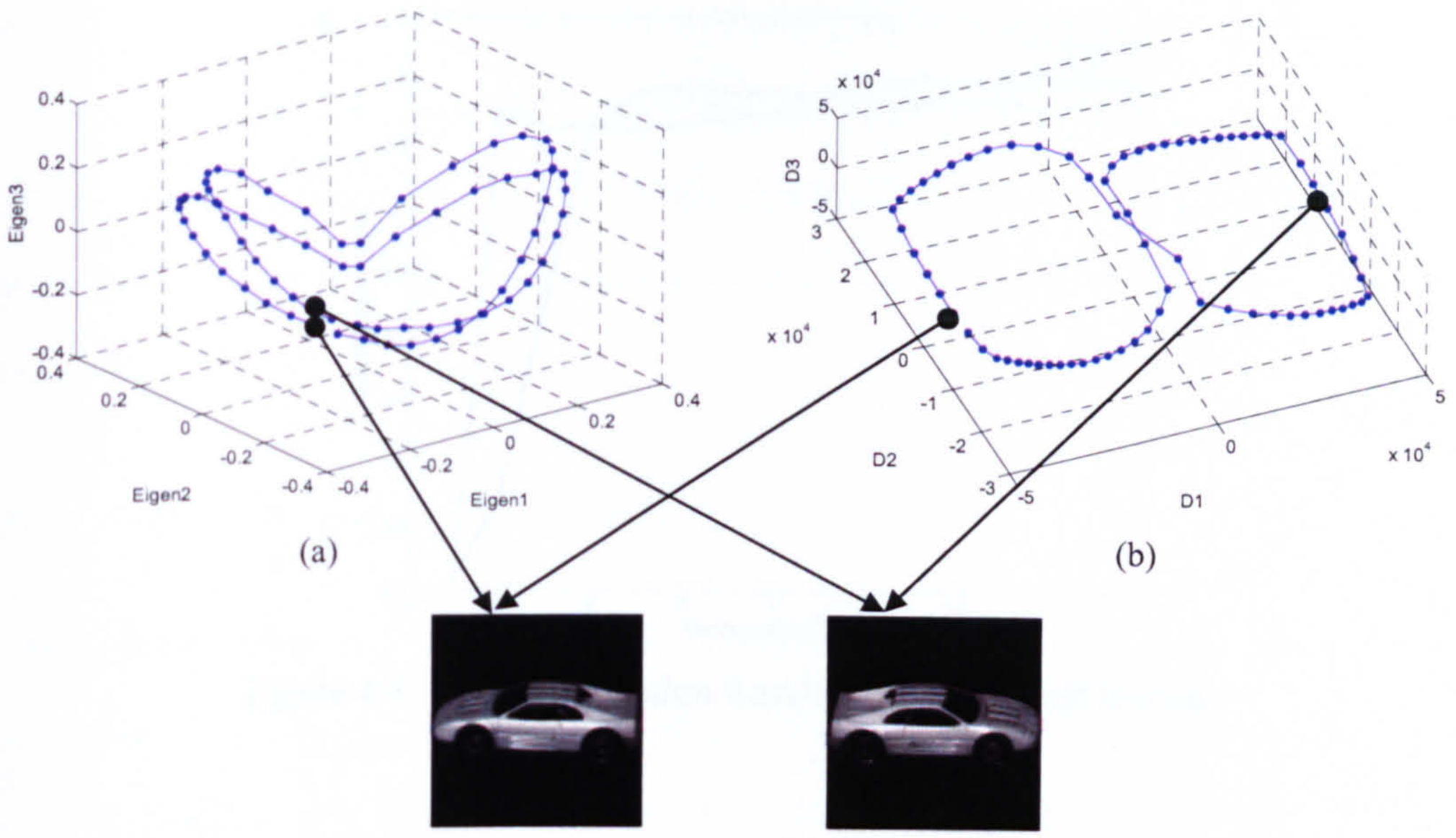


Figure 4-7 (a) and (b) are PCA based and Isomap based subspace of the image set shown in Figure 4-6

Figure 4-8 shows the identification results using PCA and Isomap. We see that in the lower dimensional subspace, the identification rate of Isomap is much better than PCA. For instance, using 2 dimensions, with Isomap, more than 85% of the input images were identified as its nearest neighbours in the observation space; while with PCA, the identification rate is below 30%. The identification rate of Isomap becomes stable at 2 dimensions while that of PCA becomes stable at 5 dimensions.

Hence, the stable identification result of Isomap is better than PCA. Again, this is because of the Isomap is a local method. In Isomap, each image is firstly assigned k nearest neighbours according to the nearest distance in the image space. In image space, each image is represented as a multidimensional vector. The dimensionality is the number of pixels in the image and the value in each dimension is the pixel value. The choice of neighbourhood is critical in this method. If the neighbours chosen according to the image space measurement is not identical to the neighbours in the observation space, the Isomap is worse than PCA. For example, Figure 4-9 shows two images from the test image set, pose 229 and pose 228. For pose 229, the 6 nearest neighbours in the observation space

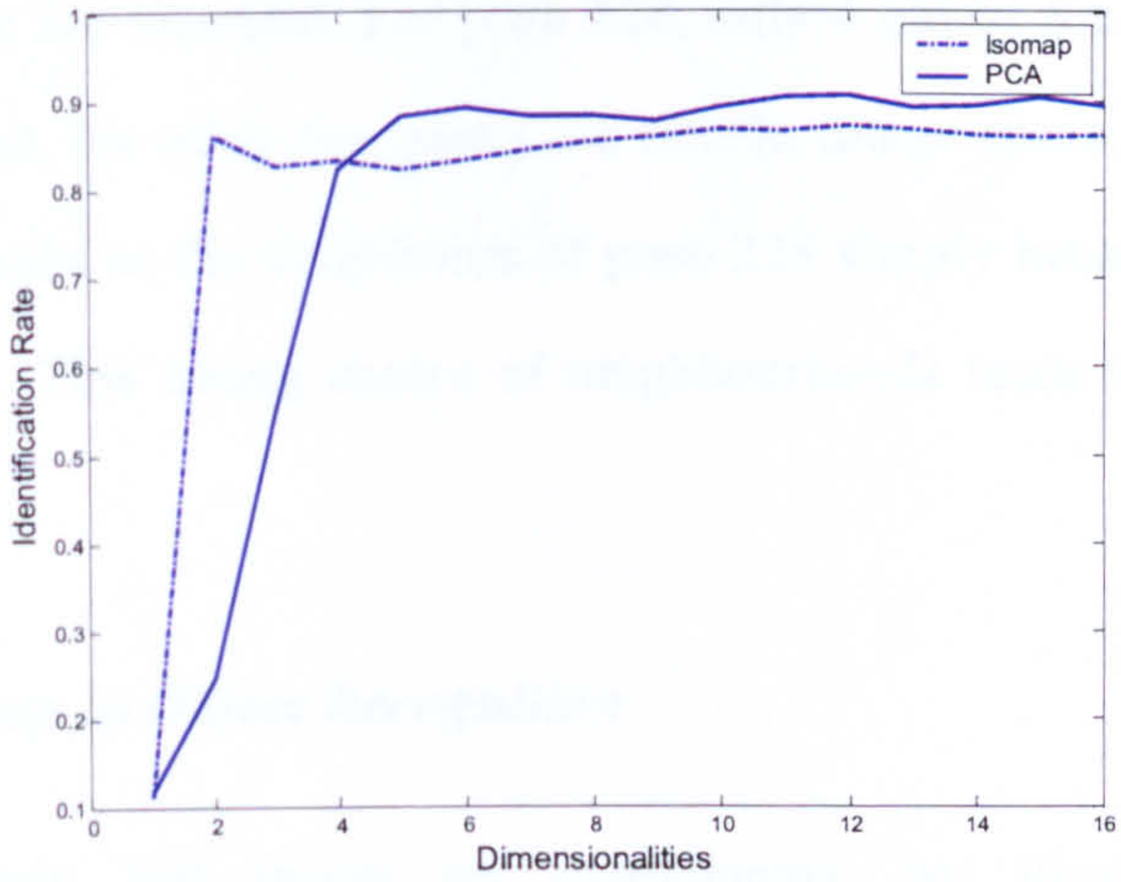


Figure 4-8 Identification Results using PCA and Isomap

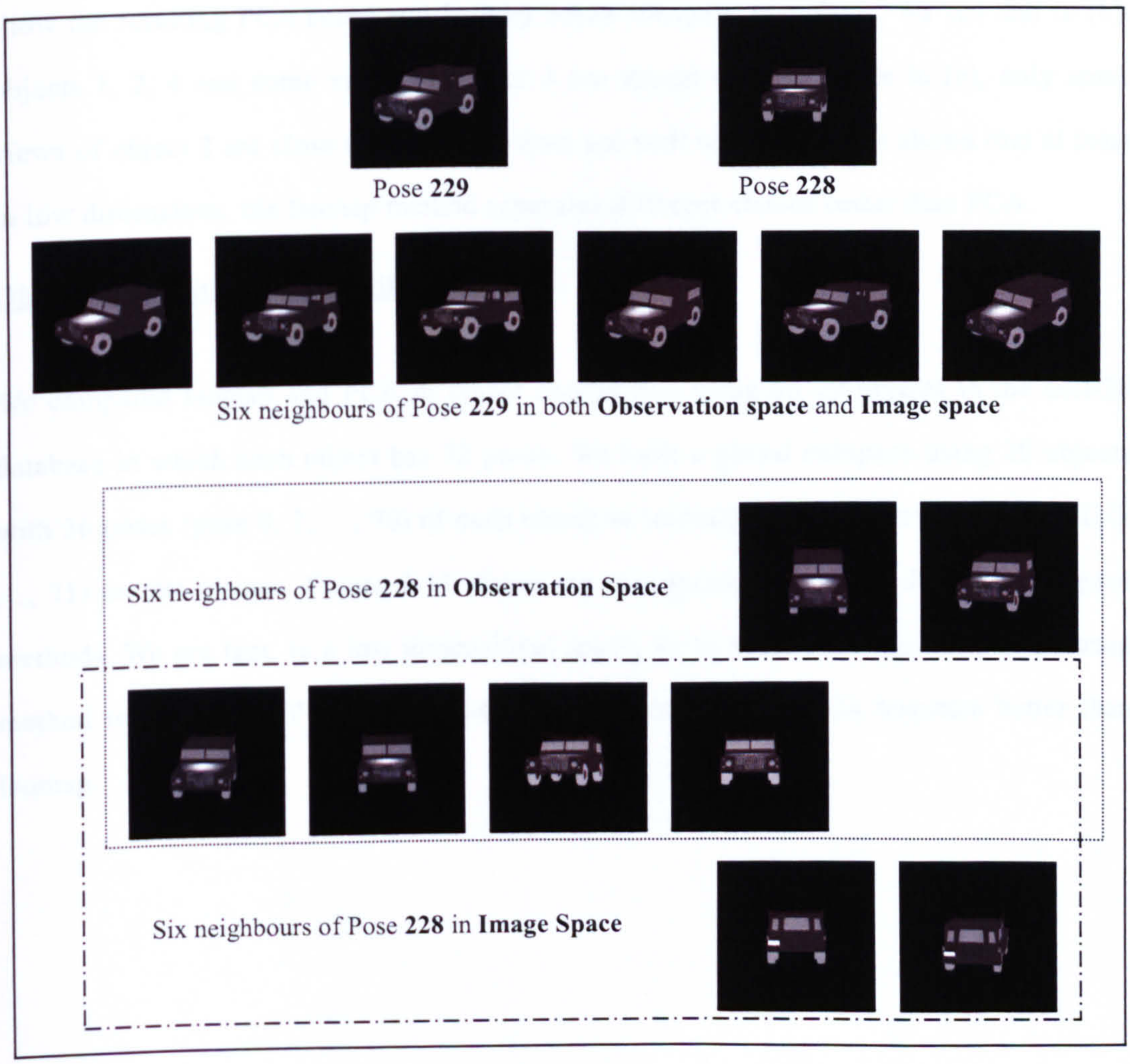


Figure 4-9 Images and their neighbours in the observation space and image space

and in the image space are identical. For pose 228, only 4 out of 6 nearest neighbours in both space are identical, the other two poses are not. In image space, the two completely different poses are chosen as the neighbours of pose 228 simply because their distance in image space are close. This wrong choice of neighbourhoods leads to a worse result for Isomap.

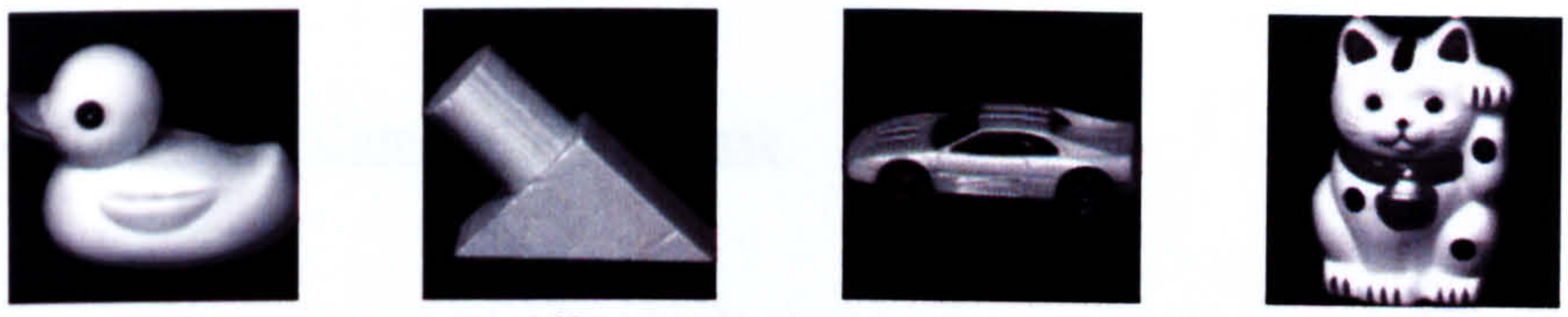
4.3.2 PCA vs. Isomap in Object Recognition

Before performing any full image set experiments, we illustrate the subspace representation of both PCA and Isomap using a small image set: 4 objects from the COIL-20 [134] database (see Figure 4-10(a)) with 72 images of each. Figure 4-10 (b) and (c) show the resulting PCA based and Isomap based subspace manifolds. We see that in (b), objects 1, 2, 4 and some views of object 3 are mixed together while in (c), only some views of object 2 are close to object 4, others are well separated. This shows that at least in low dimensions, the Isomap method separates different classes better than PCA.

Object Recognition using Coil20 database

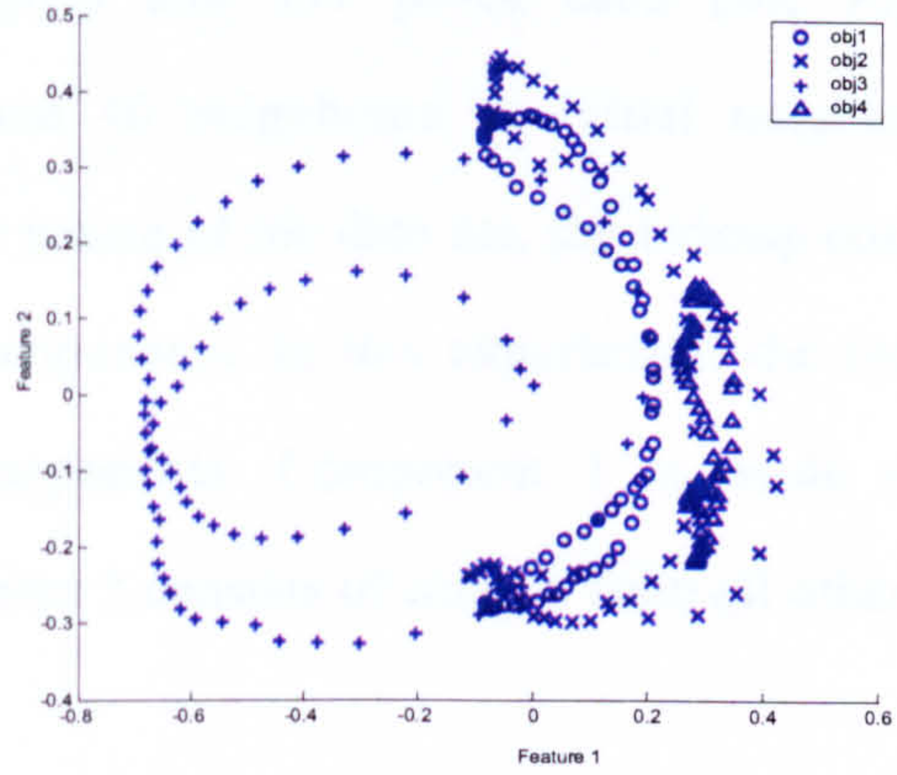
We compared Isomap and PCA in object recognition using all 20 objects in the coil-20 database in which each object has 72 poses. We built a global subspace using 20 objects with 36 poses (pose 0, 2, ..., 70) of each object as training set and other poses (pose 1, 3, ..., 71) as test images. Figure 4-11 shows the recognition rate using different subspace methods. We see that, in a low dimensional space, up to dimensionality of 6, the Isomap method is better than PCA. As we use more dimensions, the PCA becomes better than Isomap.

(a)



obj1 obj2 obj3 obj4

(b)



(c)

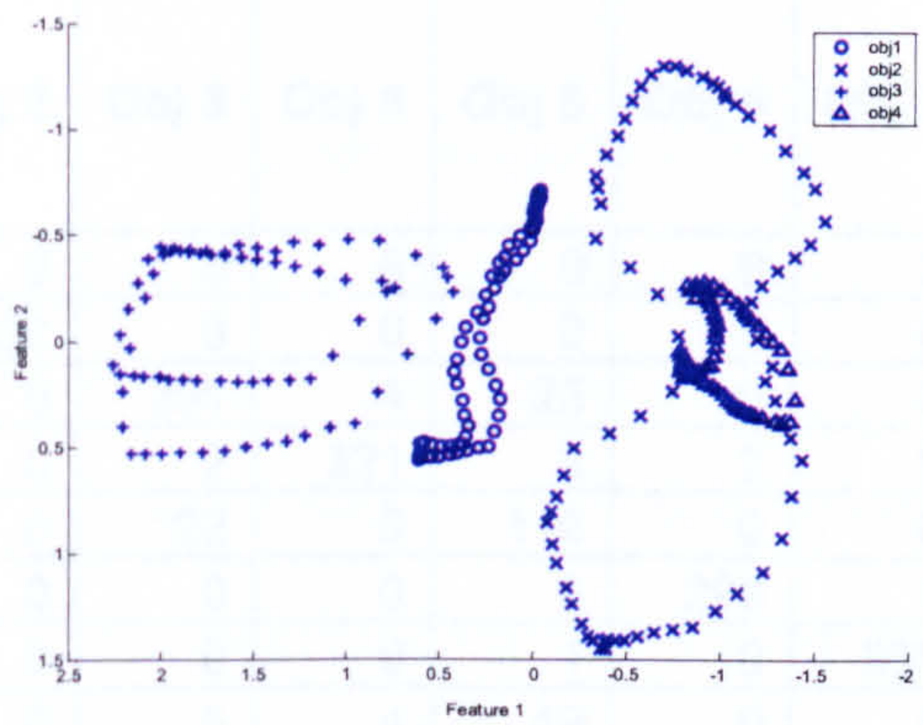


Figure 4-10 (a) the 4 objects examined (b) PCA based subspace (c) Isomap based subspace

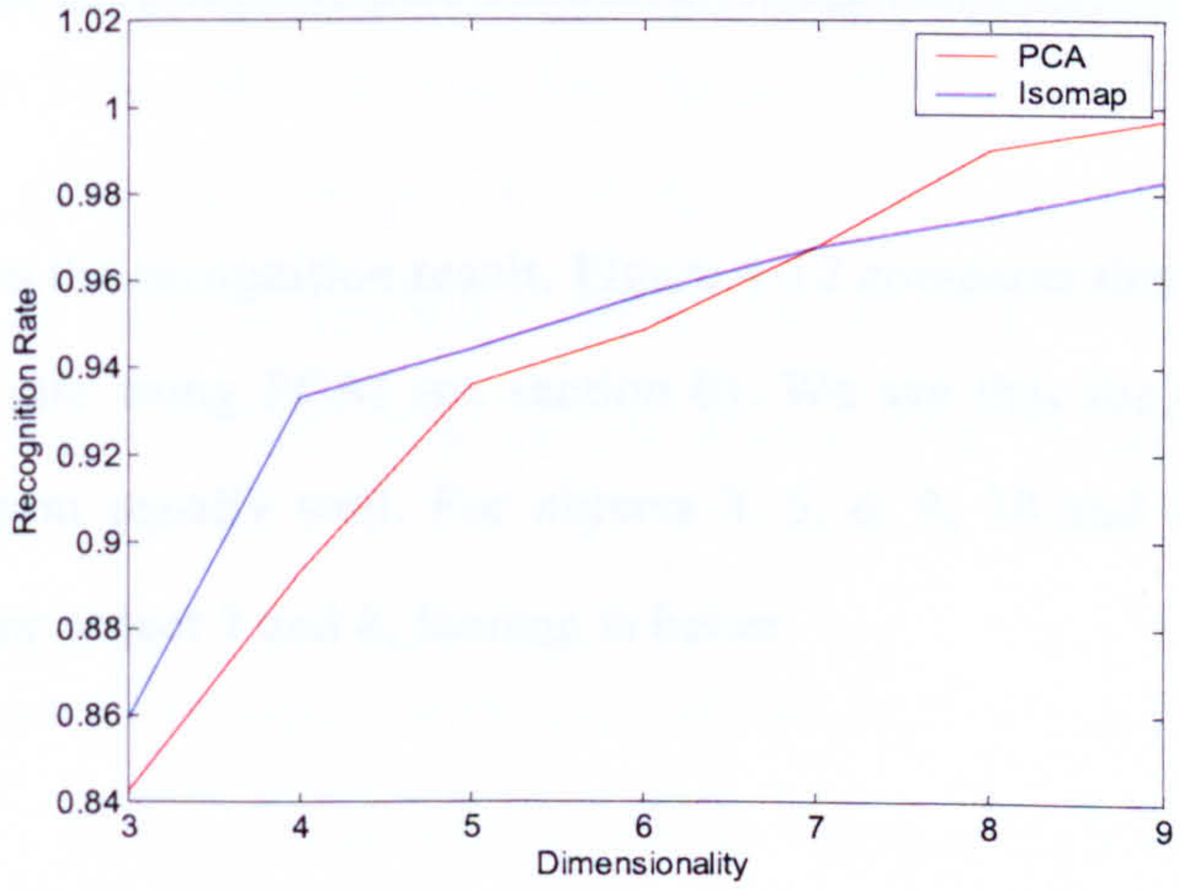


Figure 4-11 Recognition rate of PCA and Isomap subspace methods

Object Recognition using CameoSim database

In this experiment, we use Isomap to do object recognition using the CameoSim database which contains 11 objects and 337 poses each (see Figure 3-38). Here we use dimensionality of 20 and 40 neighbours as initial neighbourhoods. As discussed in Section 4.2.2, due to the nature of the data set, the Isomap could embed the whole dataset into several different components. In this experiment, the Isomap embedded the images into two connected components. Component 1 is made of all images of object 2 (Landrover) and component 2 consists of images from all other 10 objects.

Test image / Recognize as	Obj 1	Obj 2	Obj 3	Obj 4	Obj 5	Obj 6	Obj 7	Obj 8	Obj 9	Obj 10	Obj 11
Obj 1	333	0	0	0	0	0	3	0	0	0	0
Obj 2	0	337	0	0	0	0	0	0	0	0	0
Obj 3	0	0	291	4	25	0	0	1	9	0	0
Obj 4	0	0	2	321	3	2	0	0	0	2	1
Obj 5	0	0	22	6	116	0	0	59	90	0	10
Obj 6	0	0	0	0	0	299	1	0	0	20	0
Obj 7	4	0	0	0	1	0	330	0	0	3	0
Obj 8	0	0	5	4	49	0	0	149	83	4	29
Obj 9	0	0	16	1	135	0	0	121	135	4	30
Obj 10	0	0	0	0	0	35	3	0	1	303	0
Obj 11	0	0	1	1	8	1	0	7	19	1	267
Recognition Rate (%)	**98.81**	**100.0**	**86.35**	**95.25**	**34.42**	**88.72**	**97.92**	**44.21**	**40.06**	**89.91**	**79.23**

The table above shows the recognition result. Figure 4-12 compares this recognition result with the recognition rate using PCA(see section 0). We see that for object 1, 2 and 4, both algorithms perform equally well. For objects 3, 5, 6, 9, 10 and 11, PCA performs better than Isomap. For object 7 and 8, Isomap is better.

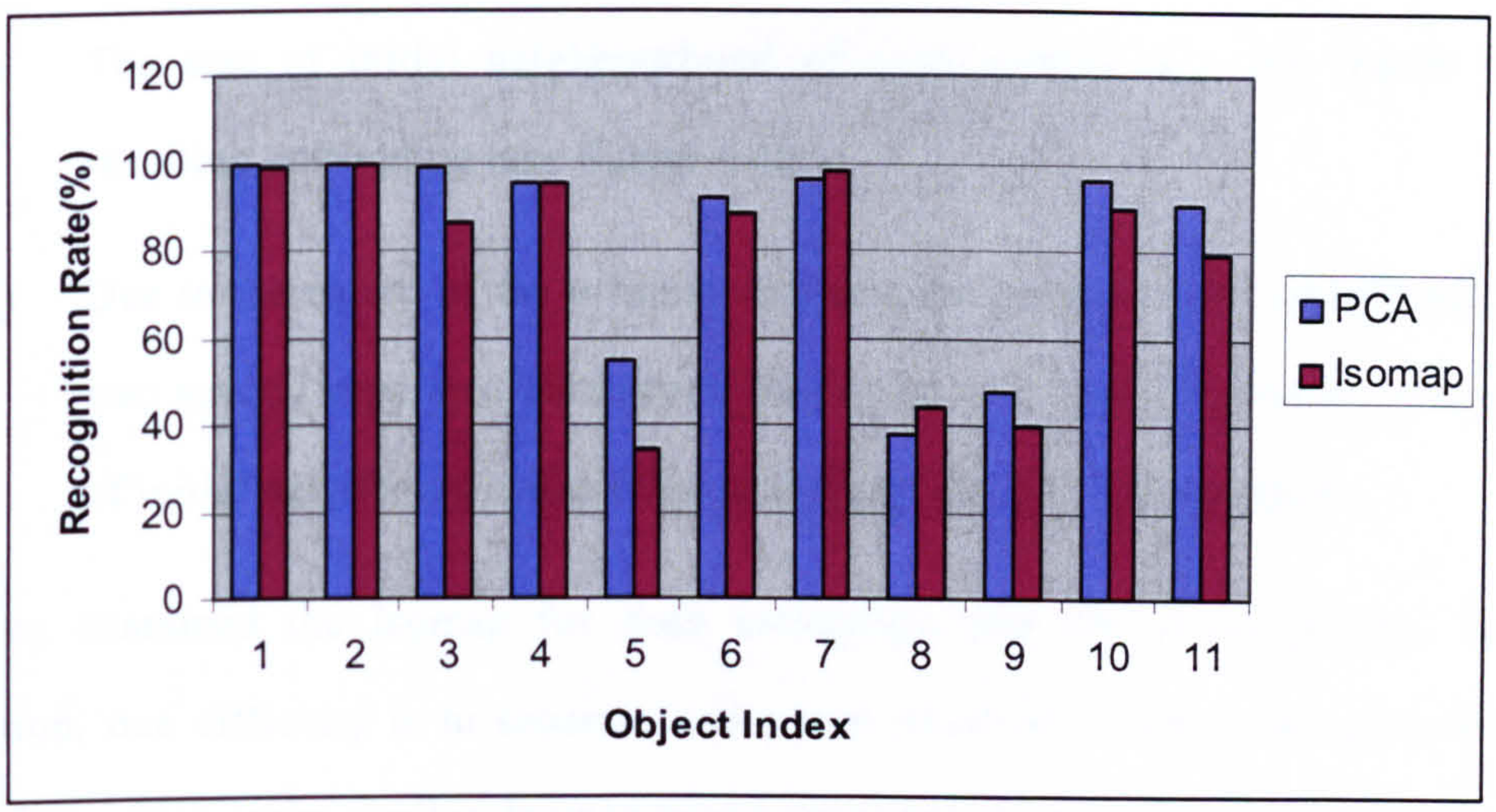


Figure 4-12 PCA v.s. Isomap in object recognition using the CameoSim database

4.4 Conclusion

In the standard appearance based object recognition method, PCA has been used as the feature extraction method. However, as a linear dimensionality reduction method, in some cases PCA is unable to find the true structure of the original data set (see the example shown in Figure 4-3). In this chapter, we have been looking at nonlinear embedding methods and examining if they can been used as the basis of appearance based recognition.

Tenenbaum etc. [127] firstly proposed the Isomap method. In their original paper, they demonstrated how Isomap could find the true structure of a set of 3D face images. Given a set of face images varying by up-down poses, left-right poses and lighting conditions, the method could automatically embed the image set into three dimension, each representing one variation (see Figure 4-1). Yang [130] has also used Isomap for face recognition. In his approach, he combined the Isomap and Fisher Linear Discriminate. The experimental results showed that this method is better than the method based on PCA for face recognition.

In this chapter, we implemented the Isomap method and applied it to the synthetic Swiss-roll data (see Figure 4-3). We see that Isomap can find the nonlinear structure of the data set while PCA cannot. We also examined two features of Isomap:

(i) The size of initial neighbourhood of each sample is a key factor for the resulting embedding (see Figure 4-4);

(ii) Due to the nature of the original data set, the Isomap could embed the dataset into several separated parts, each having its own basis. Increasing the number of initial neighbours could enhance the connection of the samples.

We then examined the Isomap for pose estimation and object recognition. In pose estimation, one difficulty is to separate two poses which are taken from quite different viewpoints which looks similar. Isomap could separated these poses while PCA cannot in 3D space (see Figure 4-7). We also tested the two algorithm on pose estimation using the CameoSim dataset. The result show that Isomap works better than PCA in a lower dimensional space but worse in a higher dimensional space. In object recognition, using both the Coil20 and CameoSim datasets, again we see that Isomap works better only in lower dimensions.

To sum up, we find that Isomap works well on relatively simple datasets where the distance between samples in their original space reflects the distance in their observation space. In more complex datasets, compared to PCA, Isomap only works better when using fewer dimensions. In applications which fewer dimensionalities is important, Isomap is better than PCA, while when the dimensionality is not a problem, then PCA could achieve a higher recognition rate. Thus, in our application, before any proper measurement of distance between images which could reflect their distance in the observation space appears, Isomap may not be a proven candidate to replace PCA in general object recognition. In next chapter, we still use PCA as the basis for dimensionality reduction.

Chapter 5

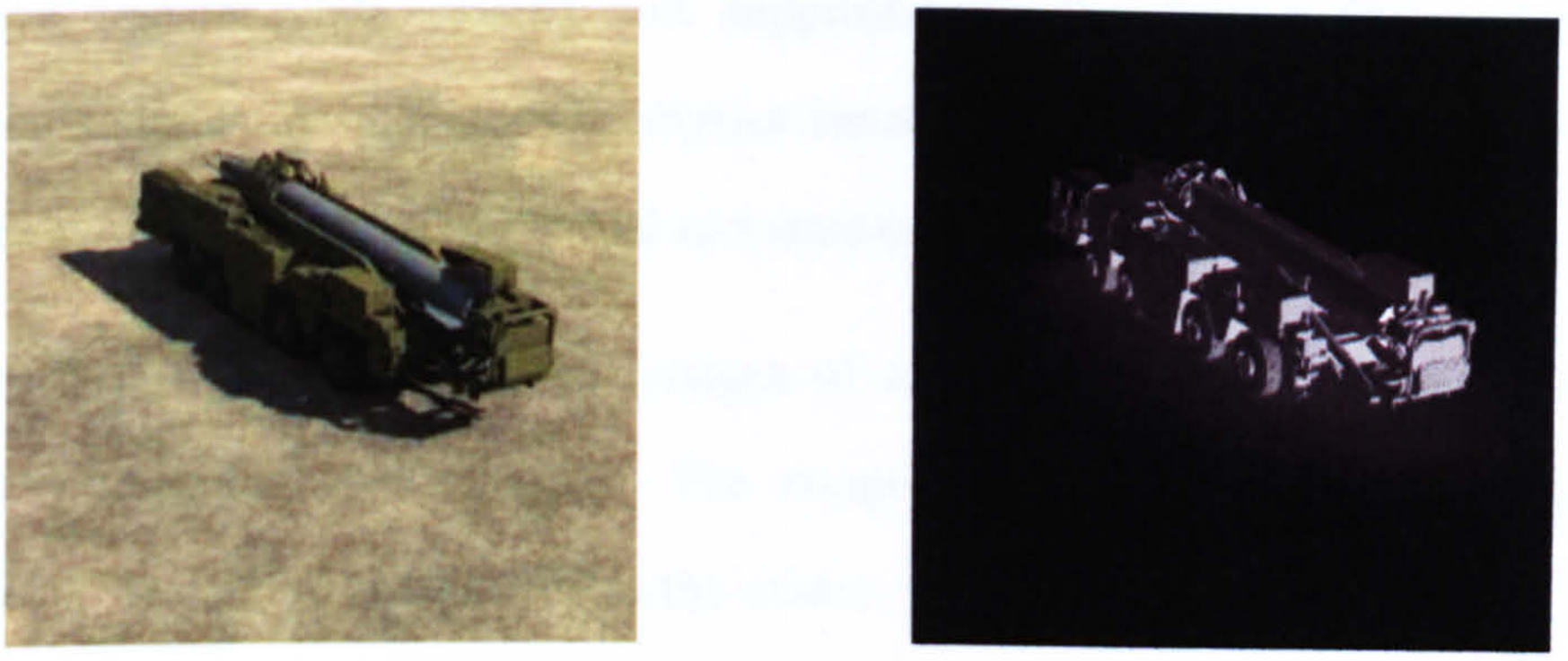
Recognition in Thermal Imagery

Object recognition in the thermal infrared spectrum has received little attention in the literature compared with recognition in visible-spectrum imagery. Socolinsky [13] has compared the performance of face recognition in visible and infrared imagery and reveals that under many circumstances, using thermal infrared imagery yields higher performance. As discussed in Chapter 2, Michel and Nanhakumar et al. [82] [83] [84] defined several thermophysical invariants for object recognition. Features are defined such that they are functions of only the thermophysical properties of the imaged object. However, practical use of their approach requires searching all the possible features for the best separation and it is not clear that a solution will always exist for a collection of object classes.

In this research, we consider several possible thermal variations of an object together with other variations such as pose using subspace methods. The uniqueness of the training method we propose is that it is able to predict subspace representations of new unknown thermal states. Thus, with this method, we should not only recognize objects in a thermal state that has been modelled, but also recognize objects in a new unknown thermal state. In this thesis, we consider the far-infrared (8-12 μm) wavebands.

5.1 Modelling Thermal Variation

It is unquestionable that a highly representative training set is desirable for the training part of the subspace methods. However, for infrared imagery, it is very difficult to gather



(a) (b)

Figure 5-1 Scud missile launcher imaged with full radiosity in the (a) visible (b) far-infrared (8-12 μm) wavebands

such a training set by taking images of real targets in real scenes, considering the possible variations due to ambient temperature, vehicle usage history and background formation, etc. Instead, we use simulated infrared scenes and try to address all possible variations in infrared imagery.

In order to provide a systematic range of image data for both the training phase and the recognition evaluation we have sought a realistic scene simulation package. Such an approach has the advantage that we can change many of the variable parameters (e.g. pose, environmental) at will, without the necessity to run and re-run field trials, which would be required to obtain real data from cameras.

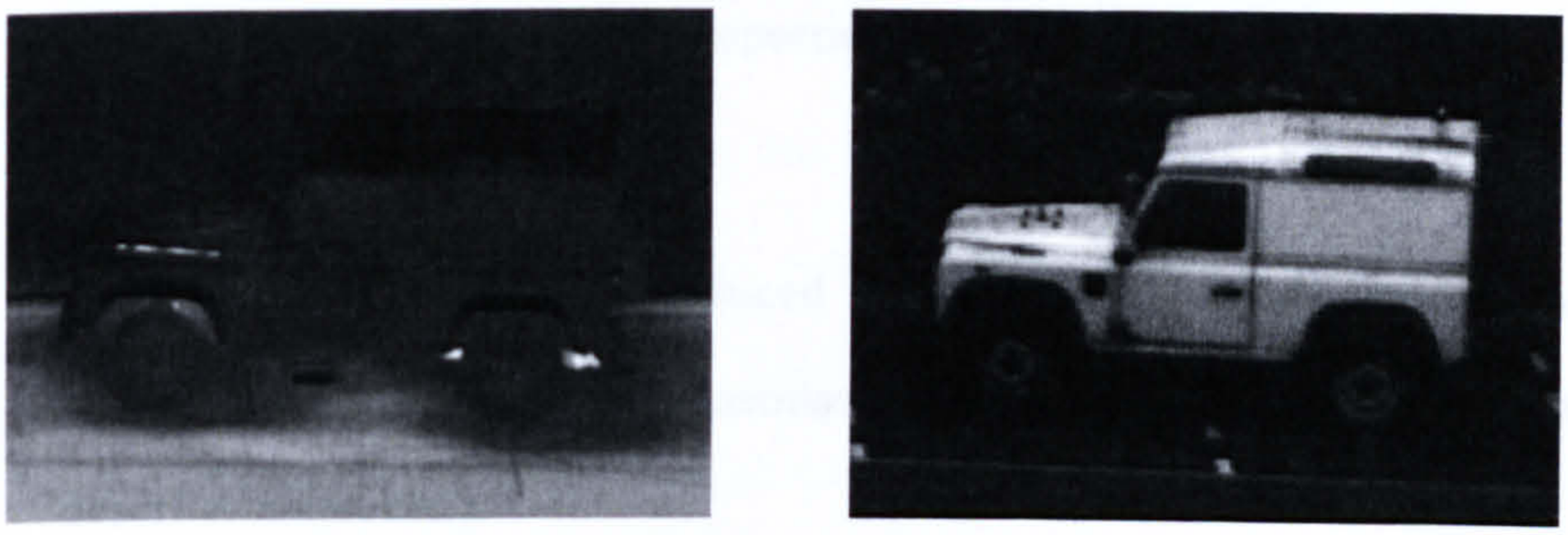
We use CameoSim [118] [119] [120], developed by Lockheed Martin UK INSYS Ltd. (http://www.insys-ltd.co.uk), to generate our simulated infrared images[11]. The software was initially developed for the UK Ministry of Defence as a synthetic scene generation tool for assessing the effectiveness of air vehicle camouflage systems. Radiance imagery can be determined at wavelengths between 0.4 and 14 μm of the electromagnetic spectrum, covering visible and infrared radiation. The package includes descriptions of atmospheric effects by invoking the widely used MODTRAN [135] [136] programs, an atmospheric

[11] Images provided by Matt Kitchin.

radioactive transfer code created and supported by the United States Air Force. CameoSim's simulation modules are physics based and its core features include models such as a ray tracing kernel and thermal and atmospheric engines.

In Figure 5-1, we demonstrate two images of a scene taken at different wavelength modalities, generated by CameoSim. The images are generated for the latitude and longitude of Stockholm, at 10:00 am in the winter, with no precipitation.

Figure 5-1 (a) depicts the scene as seen in the visible spectrum. Thermal emissions are of small consequence at these wavelengths. Figure 5-1 (b) depicts images within the far-infrared atmospheric windows (8-12 μm). We see that the appearances of the same scene in the two wavebands are quite different. For example, in Figure 5-1 (b), the grass terrain is less obvious and there is no longer a shadow. This is because the temperature of the grass is much lower than that of the body of the vehicle and the far-infrared radiation is mainly dependent on the temperature of the surface (see Appendix A). In the main body of the vehicle, we see that some dark parts in the visible images become bright in the Infrared image, e.g., the tyre. This is because the visible image is captured by sensing the surface reflection of the sunlight while the main source of thermal radiation is thermal emission which mainly depends on the temperature.



(a) (b)

Figure 5-2 Example of visible and IR images

Figure 5-2 shows an example of real infrared and visible images of a landrover. The infrared image was captured by a CEDIP JADE MWIR camera. The camera is 320x240 pixels with a specially designed IR lens. Sensitivity: The camera has an NETD of about

50mK. The visible image was captured using the PULNIX 8-bit digital visible camera. The PULNIX camera is a 768x472 pixel camera. It has a Cosmicar/Pentax lens attached to it. (Code: C31204) with varying focal length from 12.5-75mm, with F/1.8.

The visible camera captures the reflected sun radiance. So we see that the most bright part of the visible image is the main body of the car where the bright colour reflects most of the sun radiance. The IR camera responds to the emitted thermal radiance. In the IR image, the two most bright parts are the engine and the exhaust, which are the two hottest parts in the car. In the background, we see that the there is a clear boundary between the ground area and tree area, the former is much brighter. This is also due to the temperature distribution.

Hence, infrared cameras passively sense objects via the infrared radiation they emit or reflect. In the bands we are concerned with, particularly the 8-12 μm band, emitted radiation is the dominant effect. The radiation emitted depends upon the thermodynamic state of the object and its emissivity properties (for details, see the fundamentals of infrared radiation in Appendix A). For vehicles in general, the thermal variations are generated by [135]:

(i) Target conditions: exhaust grid and gases, crew compartment heating/cooling, power generator, material properties, camouflage, and target location and orientation;

(ii) Environmental variations: Induced weather (sun, clouds, rain, snow, etc.), Atmospheric influences on transmission and Geographical location (moderate, desert climate, etc.).

To model thermal variation, the thermophysical approach [82] [83] [84] generates invariants from the principle of conservation of energy at the surface of the imaged objects (see the detail in section 2.4.2). In this model, the invariants are calculated using some properties of 2 sets of points which makes it difficult to get the true value, e.g., surface temperature, ambient temperature, angle between the direction of irradiation and

the surface normal, etc.. In addition, to find the proper two sets of points, this method has to be combined with some geometric feature detection procedure. Error in feature detection could make the recognition fail. Furthermore, in this approach, the external radiation is considered as the only source, and the internal source is ignored.

As an appearance based method, we model the appearance changes caused by thermal variation. This is done by generating examples of different poses, different grids, gases and power states, etc., using a representative variation. For our simulated infrared image set, we divide the variation into 2 styles: ***single-part variation*** and ***multi-part variation***. ***Single-part variation***: Consider a set of images from the same pose and same object. We take a landrover for example, the only difference between these images are the brightness of their engine. The combination of single-part variation caused by brightness change of two or more parts of the object in the images, e.g., all possible combination of engine's states with exhaust's states, forms the ***multi-part variation***. In the experiments in this thesis, we use a fixed atmosphere condition whose parameters are as follows: Spectral atmosphere: none; Thermal atmospheric: location: Stockholm, season: winter, time: 00.00. The same framework can be used under different atmospheric conditions.

5.2 Assessment of the Simulation software – CAMEO-SIM

Unlike other simulation packages which are ray tracers only, e.g., *Opus Studio* from *Opticore*[138] and *TrueSpace*[139], or merely thermodynamic modelers such as *Radtherm*[140] and *Sirus* from *BAE SYSTEMS* and *Sowerby*, the advantage of CAMEO-SIM is that it is able to model both reflected and emitted radiation from a target and scene, incorporate wavelength dependency and scintillation if the source is monochromatic, and to model transmission through the atmosphere. In this section we expand upon the purpose and characteristics of CAMEO-SIM, in order to illustrate its scope and the reason why we chose to use it.

The purpose of the CAMEO-SIM system is to produce synthetic, high resolution, physically accurate radiance images of target vehicles in operational scenarios, at any wavelength between 0.4 and 14 μm, that includes the visual and infra-red bands. Figure 5-3 shows the components of the system. A detailed description of the system can be found in [137]. Here we only describe briefly the Target Module and the Renderer.

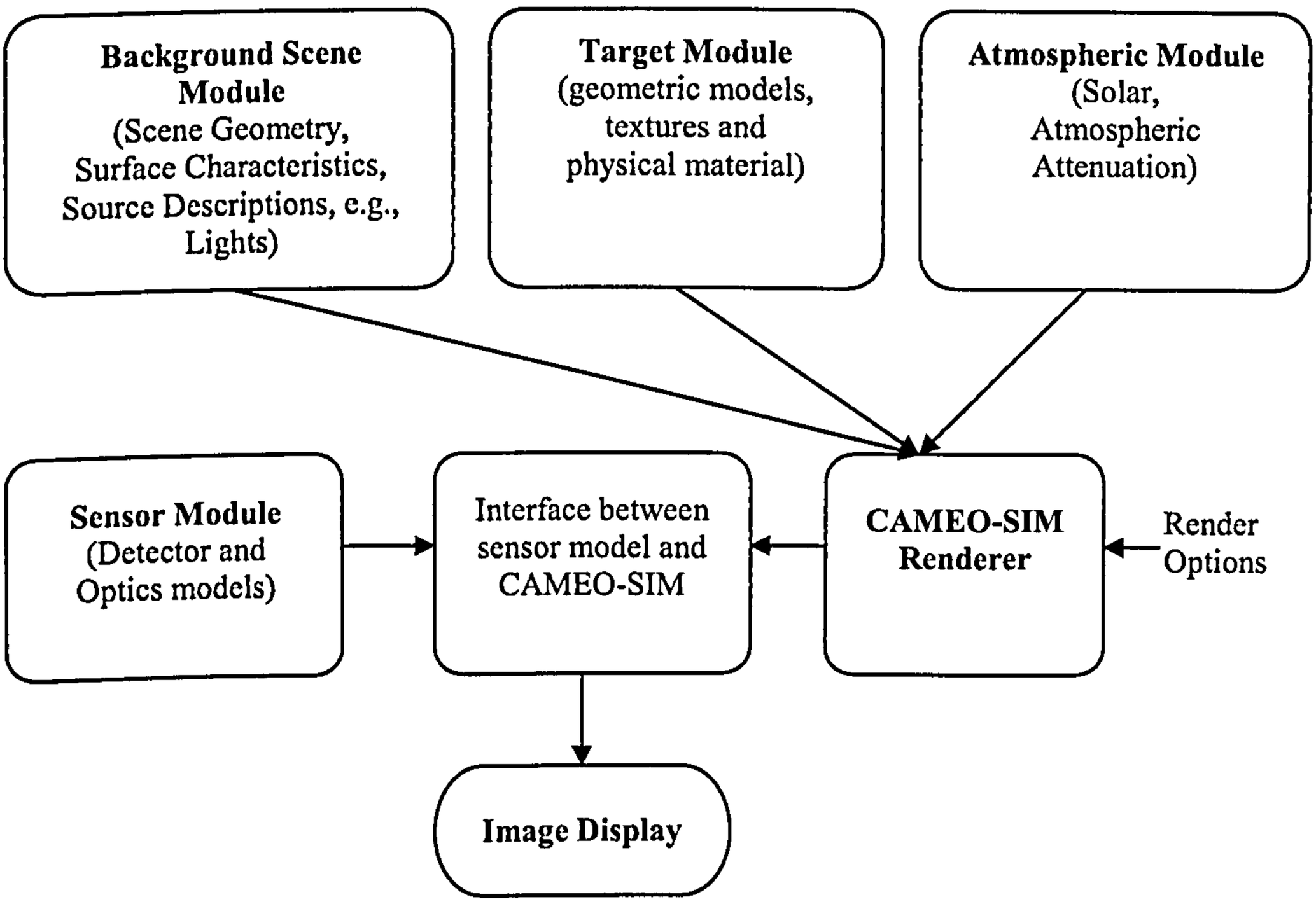


Figure 5-3 Schematic diagram of CAMEO-SIM

The 3D targets are generated from three blocks: geometric models, textures and physical material. The physical material descriptions are made up from both thermophysical and optical descriptions. The optical properties are used to define the spectral and directional properties of the surface at the given point on the 3D object based on the local curvature. The thermophysical properties, including solar absorptivity, thermal emissivity, the conductivity, the density, the specific heat, the thickness and the transpiration of the material, are used to solve the heat transfer occurring in the scene. The image generation process is now summarised. The 3D geometric models are obtained either from one of CAMEO-SIM's own model database or imported from external sources. After providing

such models from the CAMEO-SIM program, material properties are applied from databases. These contain optical and thermophysical parameters such as spectral reflectivity, conductivity etc.. These can be 'painted' on facet by facet. Texture models are also included.

Cameo-Sim has a two-pass rendering kernel that can produce high quality image streams using BRDF capable radiosity and ray-tracing algorithms. The first pass models the radiative transfer between extended surfaces. The second pass is a ray tracer to model the effect of point sources. CAMEO-SIM computes the radiance in user specified subbands for each pixel in the image. These subband radiance images can then be summed to produce an in-band radiance image.

At IR wavelengths, CAMEO-SIM uses full hemispherical integration of the incident irradiance, which enables it to account for the radiative interaction between different surfaces. The figure below shows the advantage of the full hemispherical integration over the approximation to the radiation transport equation commonly employed in simulations where target-scene interactions are ignored and only direct atmospheric illumination is accounted for. In Figure 5-4, both (a) and (b) shows the aircraft under the same conditions. However, in (b), the scene interaction is taken into account while in (a) it is not. We see that in (b), the underside of the aircraft is in positive contrast to the sky background. This is due to the incorporation of both earth thermal reflection and albedo terms.

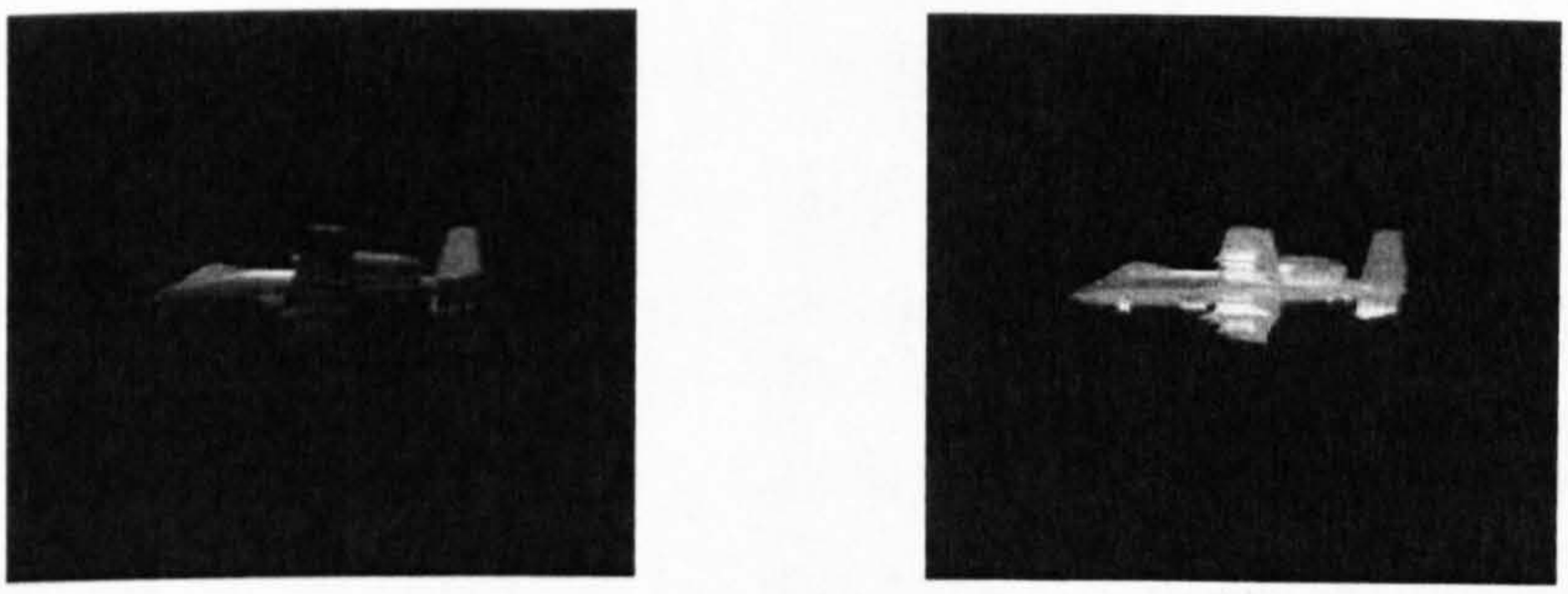


Figure 5-4 Simulation of an image of an aircraft in the mid-infrared band: (a) predicted appearance without scene interaction and (b) predicted with scene interaction (sources from [118])

A range of verification tests has been conducted by the developer to exercise different elements of the rending equations implemented within CAMEO-SIM, compared against the radiance calculated analytically. These tests include:

- Blackbody Radiance Tests – to ensure that the blackbody radiance is calculated correctly;

- Calculation of Shadowing and Blocking – to ensure that the blocking and shadowing algorithms are working accurately. Blocking and shadowing are two rendering process. The first ensures parts of the object which are not visible to the observer due to obstruction by another part are correctly accounted for; the second ensures parts of the object do not reflect the point sources if they are obscured from it by other parts;

- Spectral Calculations – to verify that the spectral integrations are being calculated accurately;

- Directional Emission of Uniformly Textured and Heated Spheres – to verifies that the second pass renderer is accounting for the directional emissivity correctly;

- Bidirectional Reflectivity of Uniformly Textured and Heated Spheres – to verify that CAMEO-SIM is interpreting the bidirectional reflectance distribution function (BRDF) correctly;

- Textured Heated Billboard for Testing Multiple Material Assignments on a Texture – to ensure that textures that have been classified using multiple material associations and transparency are interpreted properly by CAMEO-SIM.

Results show that the software computes the correct values for analytically tractable scenarios. Detailed results are shown in the table below and the process of the tests can be found in [120] [118].

Summary of validation test results.
All values are radiance (W m^{-2} sr^{-1}) unless otherwise stated. Source from [118]

Test	Expected result	Calculated
Blackbody radiance	Blackbody Radiance = 42.89 (8 to 12.5 μm band)	42.89
Shadowing and blocking	a. Radiance of irradiated area =5.1768	a. 5.1768
	b. Radiance of blocked area = 0.0	b 0.0
	c. Radiance of shadowed area = 0.0 (3 to 5 μm band)	c. 0.0
Spectral calculation	Centre pixel radiance (3 to 5 μm band) = 1.49	1.49
Directional emission	Slope of radiance along centreline =60.01 W m^{-2} $pixel^{-1}$	59.932 W m^{-2} $pixel^{-1}$
Multiple material assignment on a texture	Blackbody radiance = 8.975	Blackbody radiance =8.975
	Gray body radiance = 4.4875 (3 to 5 μm band)	Gray body radiance =4.4875
Bidirectional reflectivity	Illuminated pixel radiance = 2.3	2.3

5.3 Theory of Recognition in Thermal imagery

In this section, we discuss how thermal variation causes problems in object recognition and how we can cope with this using a subspace method. The problem is: as the thermal state of the object changes, its appearance changes in infrared imagery (see Figure 5-7). Thus, the Eigenspace method as an appearance based method may not recognize an object with a different thermal state from the one in training set. We aim to find an advanced strategy which takes account of the thermal variation.

5.3.1 Thermal variation in Subspace – Single part variation

Imagine we have an image *I* that consists of background area and foreground area. We keep the background stable and gradually increase the brightness of the foreground without changing the spatial distribution of it to form some new images. We call the

image set formed by these new images '***Partial Different Image Set***'. We wish to find out the 'Shape of the Manifold' of the Eigenspace based subspace method when the training image set is a ***Partial Different Image Set*** (PDIS). The PDIS is of interest because it is similar to the infrared image set with thermal variation (see Figure 5-7).

In section 3.1, we detailed how the Eigenspace approach works as a recognition method. In this section, we repeat the procedure, but on a special data set concerned with thermal variation. First, we try to prove the linear effect of a very simple Partial Different Image Set, e.g. as the engine heats up...

Definition of Image set:

There are n images $\vec{i}_x$ $(x \in \{1,2,\cdots n\})$ in the data set, where $\vec{i}_x$ is an image vector: $\vec{i} = [X_1, X_2 \cdots X_m]^T$. Each image has m pixels. Each image contain two parts: $\vec{i}_x = \{a_{co}, a_{ch}\}$. In a_{co}, each pixel is constant: $\vec{i}_x(k) = \vec{i}_y(k), k \in a_{co}, x, y \in \{1,2\cdots n\}$. $\vec{i}_x(k)$ is the k th element in the vector $\vec{i}_x$. k can also be understood as 'Position' in the image. In a_{ch}, each pixel value is changing, the change can follow some rule, e.g., $\vec{i}_x(k) = f_k(x), k \in a_{ch}$.

In each position in a_{ch} $(k \in a_{ch})$, the pixel values change following their own rules $f_k(x)$. For example, in Figure 5-5, a_{co} is the surrounding area and a_{ch} is the centre area. In this image set, all the pixel positions in a_{ch} change follow the same rule: $f_k(x) = 2x + 50$ where $k \in a_{ch}$. In more complex cases such as the image sets in Figure 5-7, f_k has different expressions for the different positions k.

Differential Image Matrix

Original images: $\vec{i}_x = \{a_{co}, a_{ch}\}$

Mean Image:

$$\bar{\vec{i}} = \sum_{x=1}^{n} \vec{i}_x \Big/ n \qquad (5\text{-}1)$$

$$\forall k \in a_{co}\ \bar{\vec{i}}(k) = \vec{i}_x(k), x \in \{1,2\cdots n\}$$

$$\forall k \in a_{ch}\ \bar{\vec{i}}(k) = \sum_{x=1}^{n} f_k(x) \Big/ n$$

Differential Images:

$$\vec{i}_{Dx} = \vec{i}_x - \bar{\vec{i}} \tag{5-2}$$

$$\forall\, k \in a_{co}\ \vec{i}_{Dx}(k) = 0,\ i \in \{1,2 \cdots N\}$$

$$\forall\, k \in a_{ch}\ \vec{i}_{Dx}(k) = \vec{i}_x(k) - \sum_{x=1}^{n} f_k(x) \Big/ n$$

Differential Matrix:

$$D = \begin{bmatrix} \vec{i}_{D1} & \vec{i}_{D2} & \cdots & \vec{i}_{Dn} \end{bmatrix} \tag{5-3}$$

Compare to the procedure in Chapter 3, the images don't sustain energy normalization. For Eigenspace based recognition in visible imagery, the energy normalization is designed to reduce the effect by variations in the intensity of illumination or aperture of the imaging system. In an ideal radiosity calibrated[12] infrared imaging system, there will be no aperture. In addition, the absolute value of the radiation is important since it reflects some properties of the surface of the object being imaged. So in this algorithm designed for infrared imagery, we don't use energy normalization.

Build Eigenspace

Form the covariance matrix C:

$$\begin{aligned} C_{1,1} &= D(1,:) \bullet D(1,:)' \\ &\vdots \\ C_{p,q} &= D(p,:) \bullet D(q,:)' \\ &\vdots \\ C_{m,m} &= D(m,:) \bullet D(m,:)' \end{aligned} \tag{5-4}$$

Note that $D(1,:)$ is the pixel values of position 1 from all images, thus,

$$\forall\, (p \in a_{co} \cup q \in a_{co}),\ C_{p,q} = 0$$

$$\forall (p \in a_{ch} \cap q \in a_{ch}),\ C_{p,q} = \sum_i f_p(i) \cdot f_q(i)$$

The basis of the Eigenspace are the $\vec{e}$ s which satisfy the equation: $C \cdot \vec{e} = \lambda \cdot \vec{e}$, where λ and $\vec{e}$ are an eigenvalue and an eigenvector of C. In C, for any row whose elements are all zeros, the same row in $\vec{e}$ must be zero.

$$\Rightarrow \quad \forall\, k \in a_{co},\ \vec{e}(k) = 0; \tag{5-5}$$

---

[12] For details about radiosity calibration, see Appendix A.

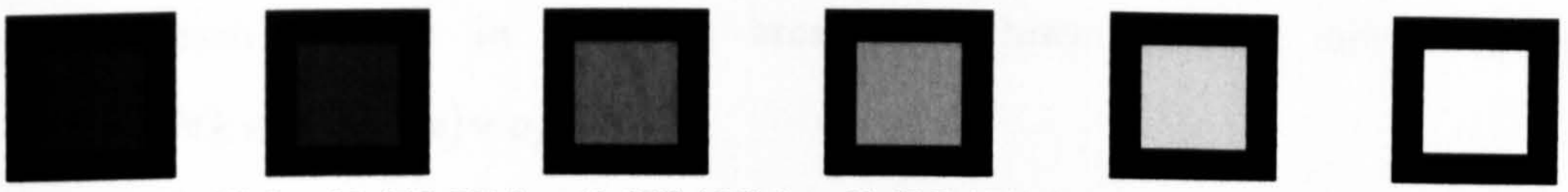


Figure 5-5 Example images of image set 1
(e.g. FG:52 means that the brightness value at the centre (foreground) area is 52)

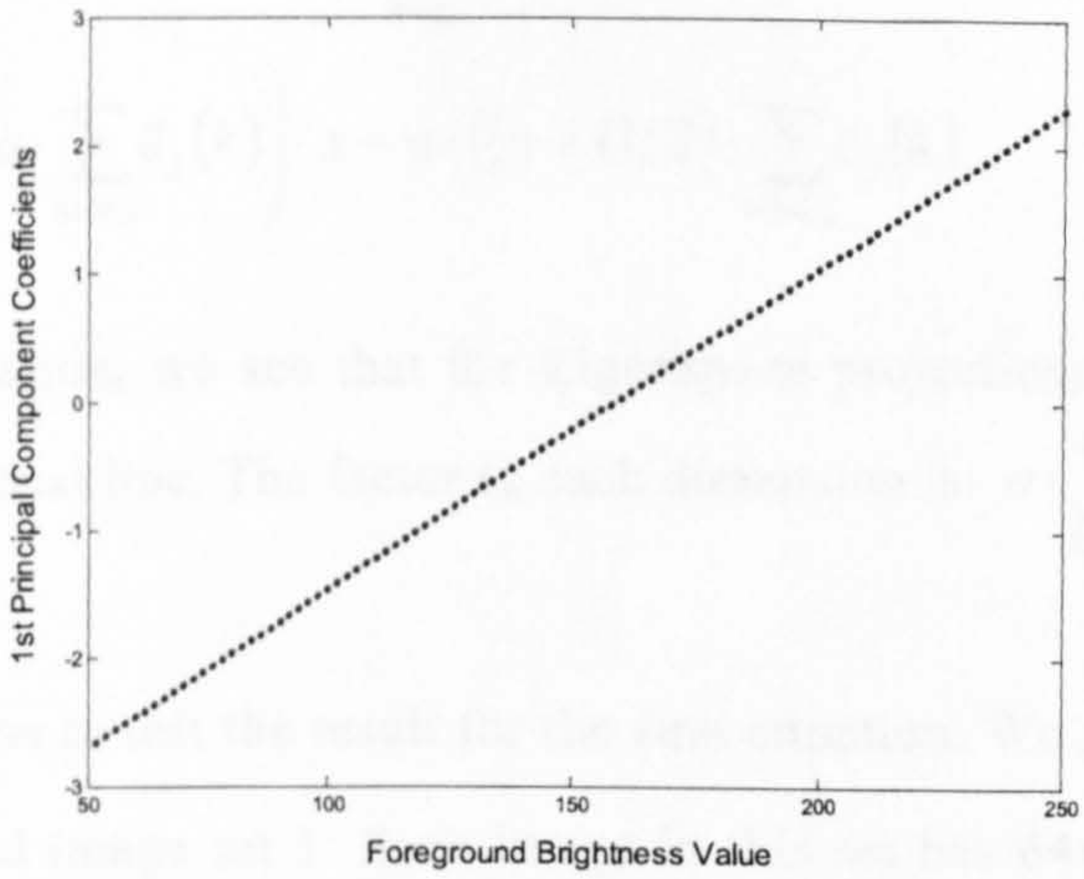


Figure 5-6 First Principal Component Coefficients over the Image Set 1.

Projection to Eigenspace

When we project the image vectors into the Eigenspace, the coefficients are calculated by:

$$c_j(x) = \vec{i}_{Dx} \cdot \vec{e}_j = \sum_{k \in a_{ch}} \left(\vec{i}_x(k) - \sum_{x=1}^{n} f_k(x) \Big/ n \right) \cdot \vec{e}_j(k) \qquad (5\text{-}6)$$

$c_j(x)$ is the j th coefficient of image $\vec{i}_x$.

Equation (5-6) is a general expression of how to calculate the Eigenspace coefficients of each image in a ***Partial Different Image Set (PDIS)***, in which x is the image index. In the following text, we will discuss two different ***PDIS*** with different $f_k(x)$ and try to find a rule for the Eigenspace projection of the three ***PDIS***:

(i) each position in the changing area a_{ch} follows the same linear equation: $\forall k \in a_{ch},\ f_k(x) = f(x) = ax + b$;

(ii) each position in changing area a_{ch} follows different linear equation: $\forall k \in a_{ch}, f_k(x) = a_k x + b_k$.

For the **first situation,** the coefficient in Eigenspace for each image can be calculated by

$$
\begin{aligned}
c_j(x) &= \sum_{k \in a_{ch}} \left(ax - a\frac{\sum x}{n} \right) \cdot \vec{e}_j(k) \\
&= a \cdot (x - (n+1)/2) \cdot \sum_{k \in a_{ch}} \vec{e}_j(k) \\
&= \left(a \cdot \sum_{k \in a_{ch}} \vec{e}_j(k) \right) \cdot x - a \cdot ((n+1)/2) \cdot \sum_{k \in a_{ch}} \vec{e}_j(k)
\end{aligned}
\tag{5-7}
$$

From the above equation, we see that the Eigenspace projections of the first PDIS will form a multidimensional line. The factor in each dimension is: $a \cdot \sum_{k \in a_{ch}} \vec{e}_j(k)$.

Now we do simulation to test the result for the first situation. We generate a simple PDIS of 100 images, called image set 1. Each image in this set has 64×64 pixels, with 40×40 foreground pixels at the centre and other pixels as background. In this image set, the brightness of the background of all images is 1, while the brightness of foreground of the first image is 52 with an increase of 2 at next image and end up with a brightness of 250 of the last image. In image set 1, the 40x40 pixels foreground is the a_{ch} area and the background is the a_{co}. Figure 5-5 shows some images of this image set and the pixel brightness of the foreground area, e.g., FG:52 means the brightness of foreground area is 52.

We use image set 1 as training images and use them to form an Eigenspace. In the Eigenspace, since the first eigenvector can explain 100% of the variance among images, we need only to use the first eigenvector in this subspace method. The variation of the first Principal Component (PC) Coefficient over the image index of Image Set 1 is shown in Figure 5-6, which is also the plot for PC Coefficients over the foreground brightness value because the later is linear with the image index. In Equation (5-7), we proved that the Eigenspace projections of the images in the firsts type of PDIS form a multidimensional line in Eigenspace. From this simulation, we see that this line is a line

along the first dimension of Eigenspace. In other words, this line is parallel to the first eigenvector. This is because the first principal component explains all the variance among the data.

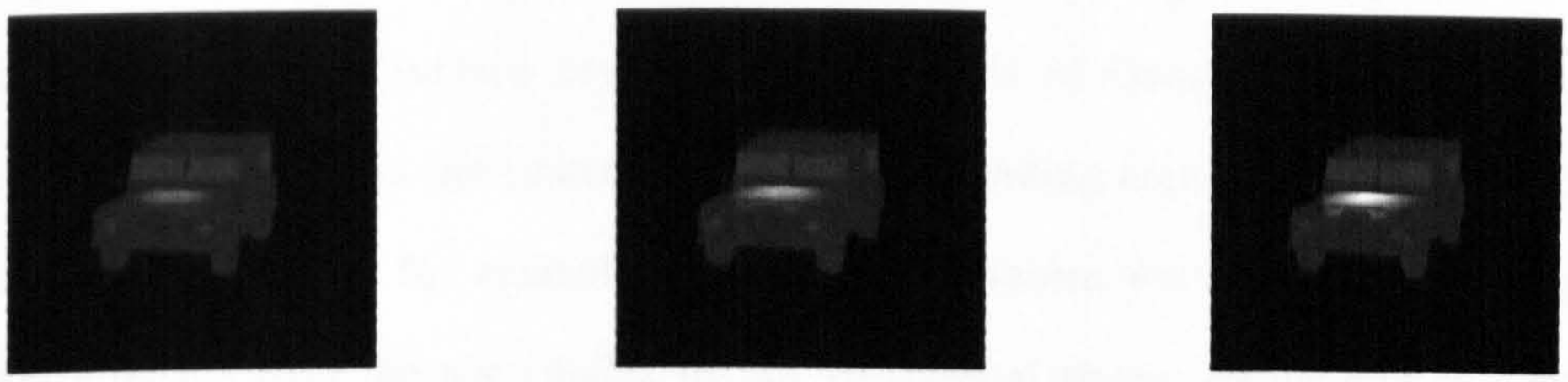


Figure 5-7 Example of infrared image set with thermal variation in Grill and Engine area (Landrover TSig1 frame-066 T1, T5 and T10)

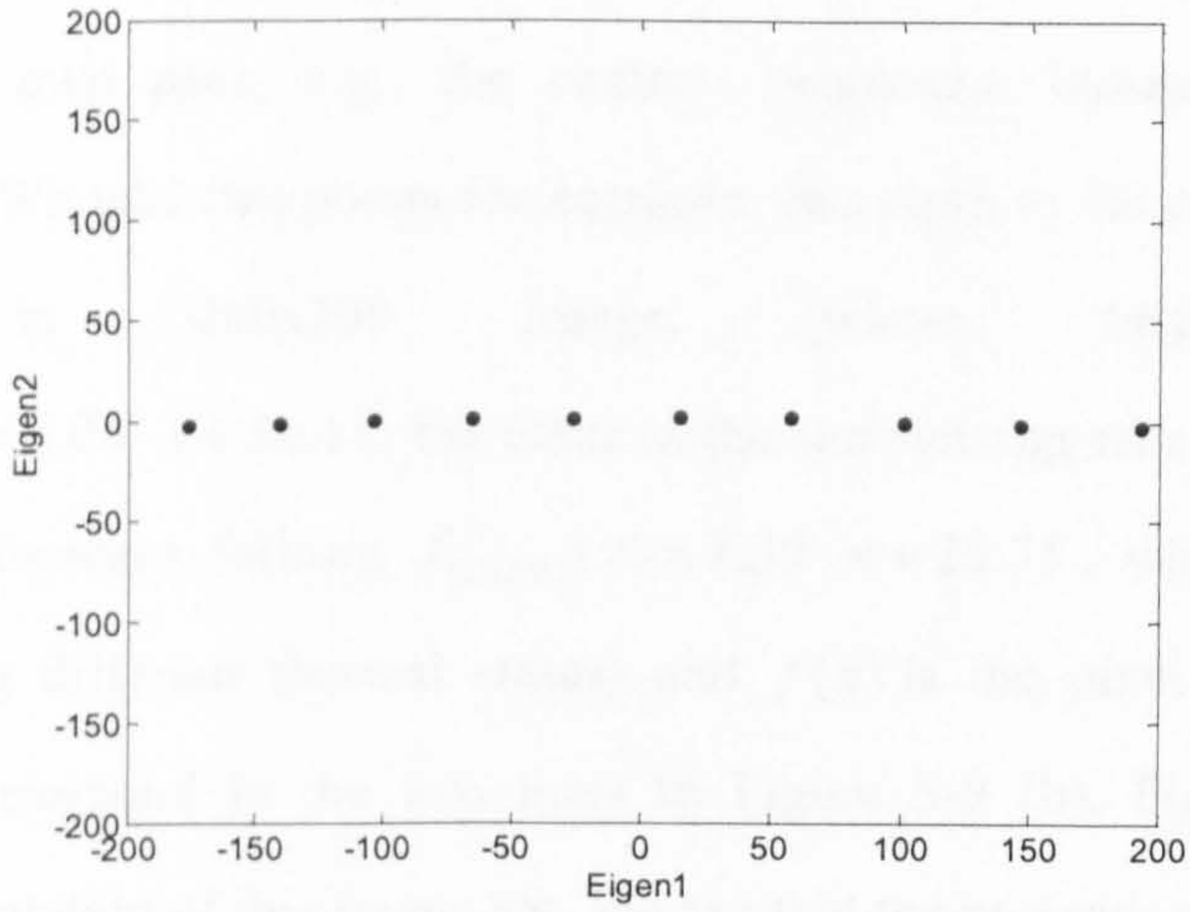


Figure 5-8 Eigenspace representation of the image set shown in Figure 5-7

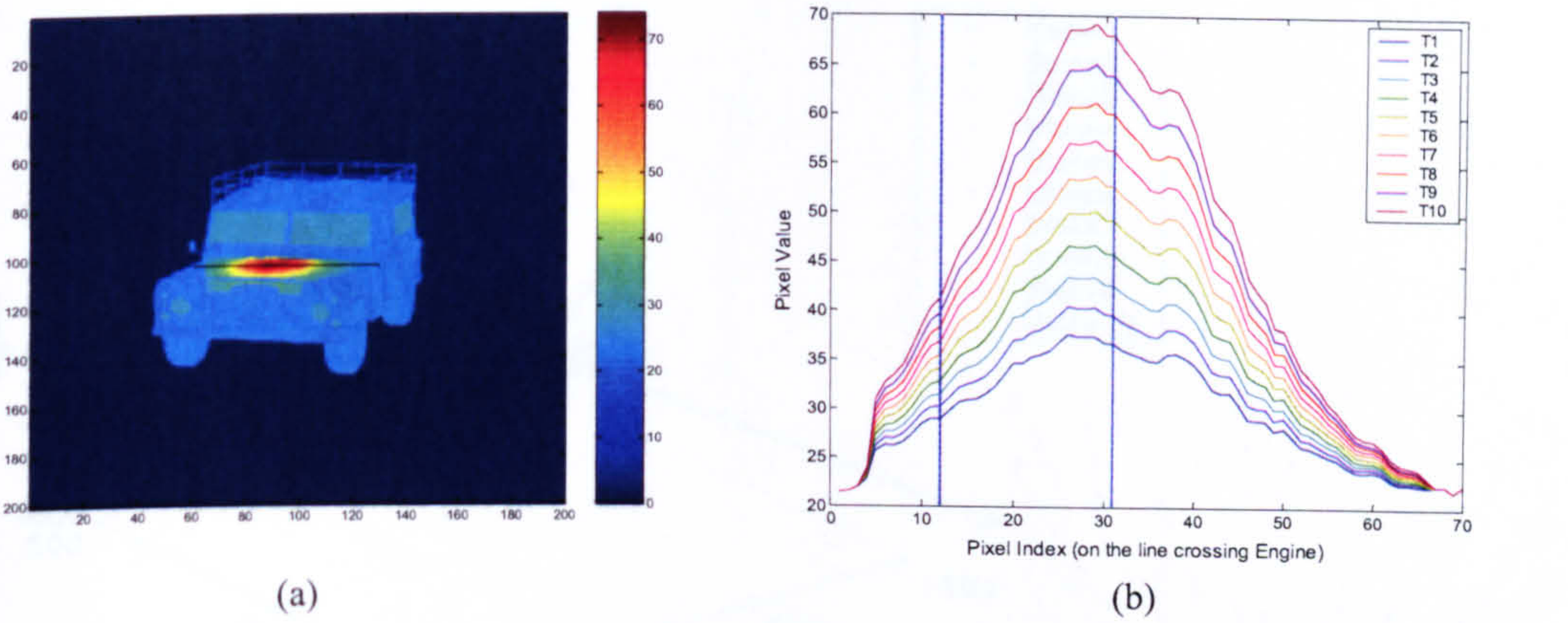
The **second situation** is a more general case of the first situation, in which the linear functions for each pixel are different. The coefficient in Eigenspace for each image can be calculated by

$$c_j(x) = \sum_{k \in a_{ch}} \left(a_k x + b_k - \sum_x (a_k x + b_k) \Big/ n \right) \vec{e}_j(k) \qquad (5\text{-}8)$$

For all x, $\sum_{x} a_k x + b_k$ is a constant, so the projections in Eigenspace will still form a multidimensional line. The slope in each dimension is $\sum_{k \in a_{ch}} a_k \vec{e}_j(k)$.

The image set shown in Figure 5-7 belongs to the **second situation**. In the image set, the engine part on the landrover is the a_{ch} area. As shown in Figure 5-9 (a), the sensed thermal radiation in the surface covering the engine is of Gaussian-shape, that is, the intensity gets smaller from the centre area to the surrounding area smoothly. We take one dimension in the surface for example, Figure 5-9 (b) shows the near Gaussian-shape of the pixel intensity over the pixel index for all 10 thermal states. As the engine heats up, the temperature of the surface close to the engine area increases.

However, the difference from the first situation is, the brightness of the whole area does not increase following the same linear equation. Instead, the brightness of each part increases at their own pace, e.g., the centre's brightness increases faster than the surrounding areas. We take two points for example: one close to the centre area of engine, (91, 103) in 200x200 image, whose brightness increase follows $f_{(91,103)}(x) = 3.04 \cdot x + 36.11$; the other in the surrounding area of engine, (72, 103), whose brightness increase follows $f_{(72,103)}(x) = 1.39 \cdot x + 29.75$, where x is the image index (representing different thermal states) and $f(x)$ is the pixel intensity. The two example points correspond to the two lines in Figure 5-9 (b). Figure 5-8 shows the Eigenspace representation of this image set. We see that the projection points are on a line in Eigenspace following the direction of the first eigenvector.



(a) (b)

Figure 5-9 (a) Thermal state T10 of Tsig3 in Landrover image set (b) The pixel intensities of a line crossing the engine (the black line in (a)) of 10 different thermal states in Tsig3.

This example demonstrates that for one pose of an object under any number of thermal states, the projection in the Eigenspace is a multidimensional straight line. For object recognition, we project MANY pose images in different thermal states into one Eigenspace to form an object Eigenspace. For example, in our training data, we have 337 poses for each object. Building the Eigenspace together with other pose images will affect the straightness of the line. However, it's not possible to mathematically model how much the straightness of the line will be affected as this depends on many factors, e.g., the geometry and surface reflectance of objects, the views chosen.

To demonstrate explicitly the form of the thermal variation, Figure 5-11 shows the Eigenspace representation of all 337 poses, each having 10 thermal states, in which the pose 66 is shown in Figure 5-7. In Figure 5-11, we see that the subspace projection of different thermal state from the same pose stay in a 'line'. However, the figure only shows the first 3 out of 100 dimensions. To test whether they really follow a 'line' in the multidimensional space, we do PCA on those groups (each group is 10 images from one pose but with different thermal state). Please note that here the PCA is not applied to the image, but applied to the multidimensional points in the Eigenspace, as shown in Figure

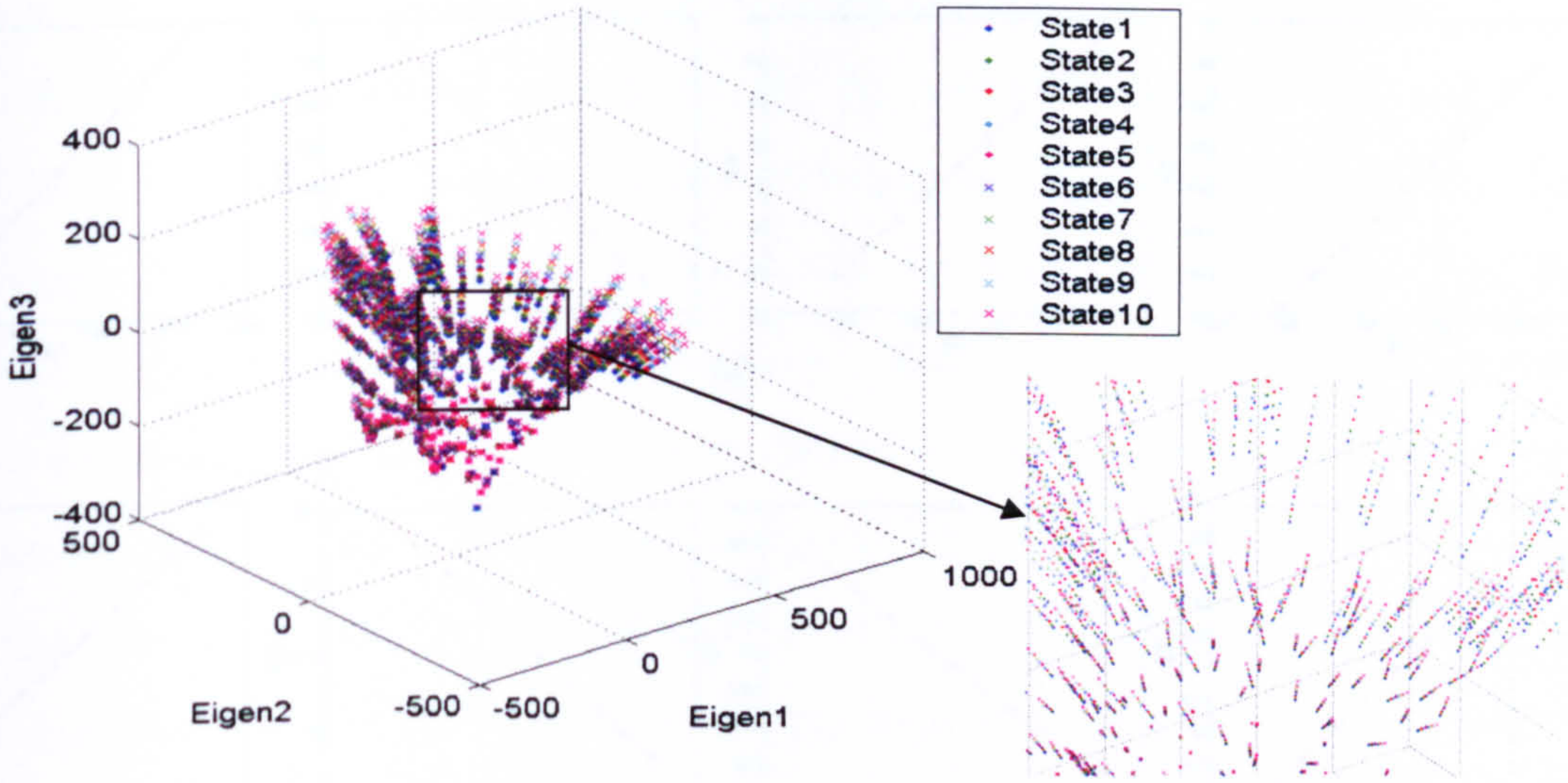


Figure 5-11 Subspace representation of Linear variation

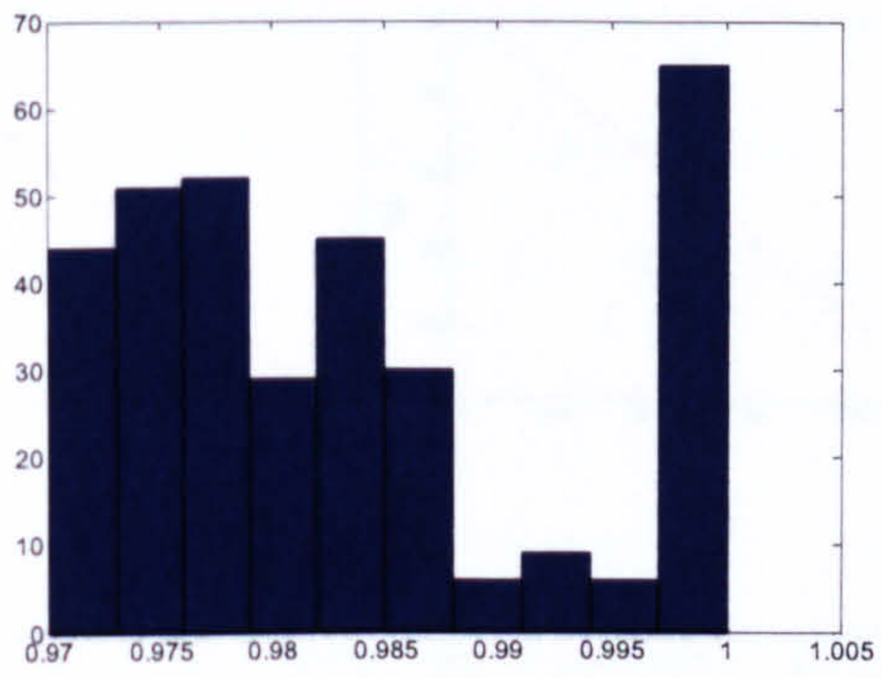


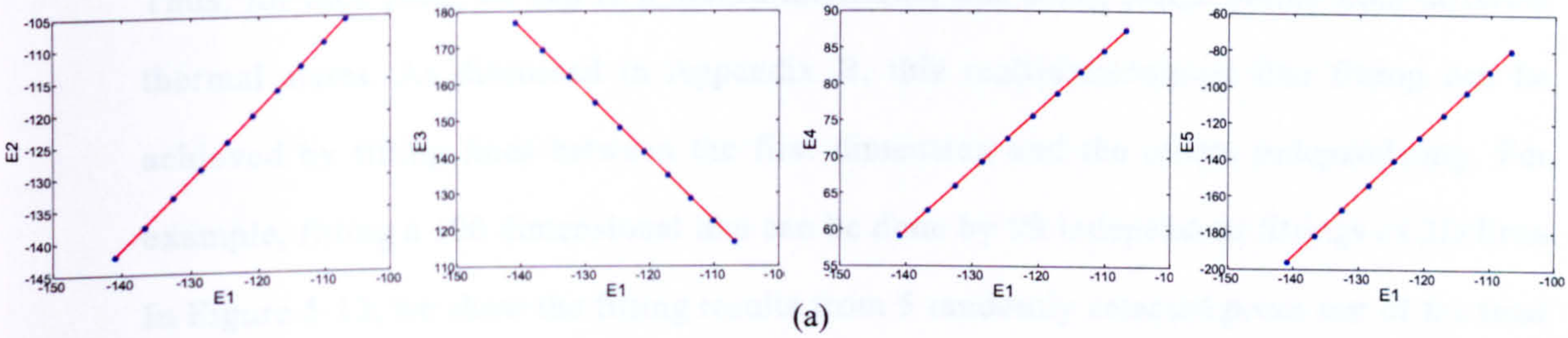
Figure 5-10 Histogram of α

5-11. There are totally 10 eigenvectors (equal to the total number of images). We did the PCA for all 337 groups and use a factor

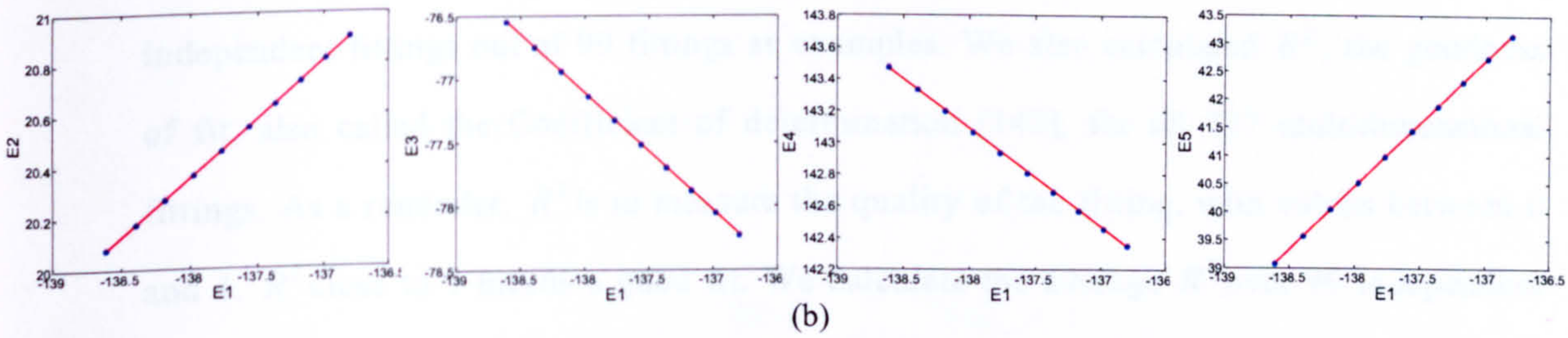
$$\alpha = \frac{value\ of\ the\ 1st\ eigenvalue}{sum\ of\ all\ 10\ eigenvalues}$$

to measure the linearity of the group points (If α is very close to 1, that means the first PC accounts for almost all the variance between the points and thus the true dimensionality of the points cloud is one. The points are considered to be almost in a line). Figure 5-10 shows the histogram of α of all 337 groups and from this we can tell that all the groups follow a line or direction.

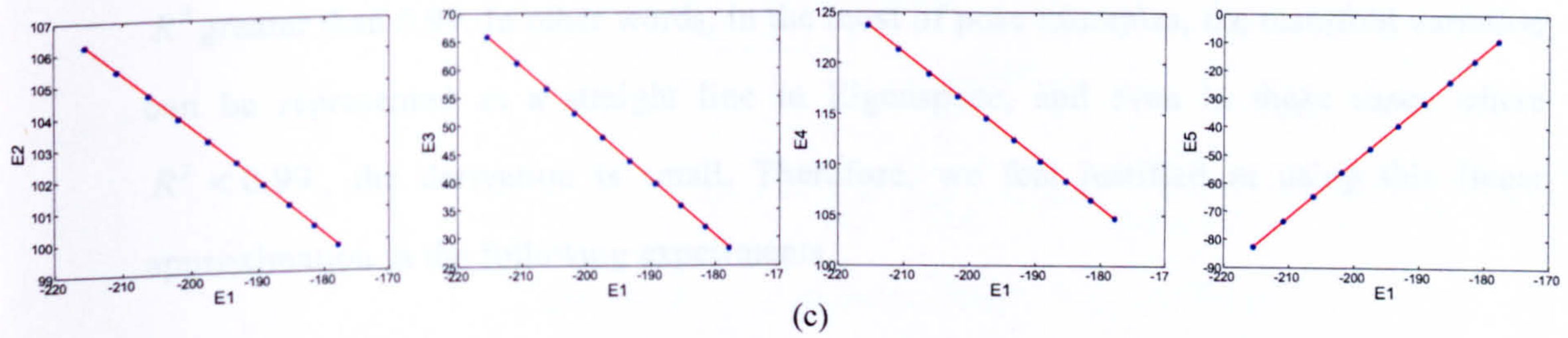
0



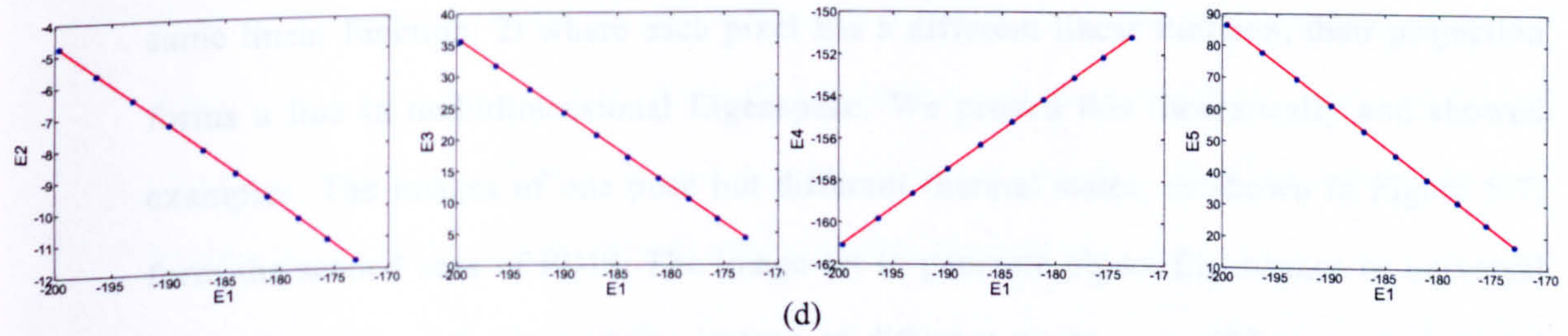
(a)



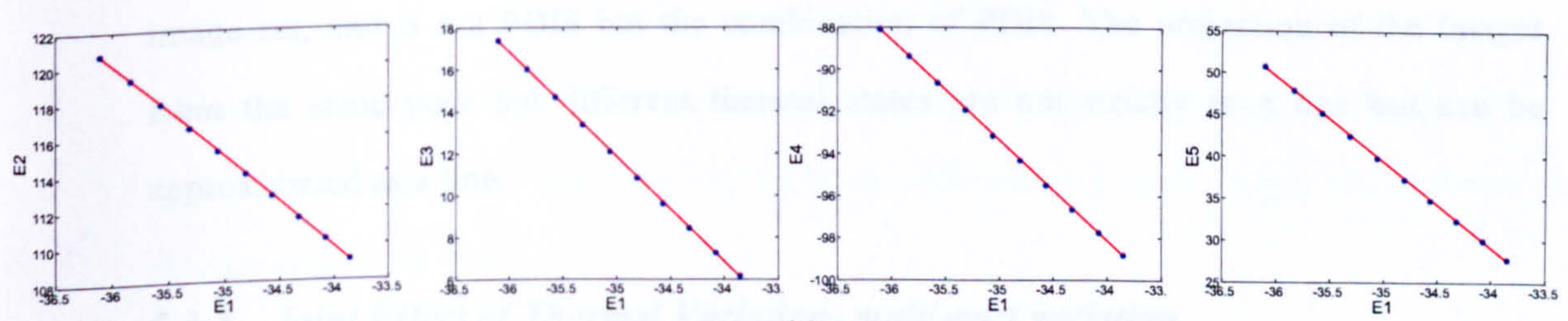
(b)



(c)



(d)



(e)

Figure 5-12 The group of four figures in row from (a) to (e) shows the multi-dimensional line fitting results using the first dimension in Eigenspace as an independent variable and the others as dependent. Left–E1&E2, second left – E1&E3, third left – E1&E4, right – E1&E5. (a)-pose 320, (b)-pose 77, (c)-pose 204, (d)-pose 163, (e)-pose 300

Thus, for each pose, we can fit a multidimensional line using image points from different thermal states. As discussed in Appendix B, this multidimensional line fitting can be achieved by fitting lines between the first dimension and the others independently. For example, fitting a 100 dimensional line can be done by 99 independent fittings of 2D lines. In Figure 5-12, we show the fitting results from 5 randomly selected poses out of the total 337 poses using the data whose whole Eigenspace is shown in Figure 5-11, using four independent fittings out of 99 fittings as examples. We also calculated R^2, the goodness of fit, also called the Coefficient of determination [142], for all 337 multidimensional fittings. As a reminder, R^2 is to measure the quality of the fitting, with values between 0 and 1. R^2 close to 1 means a good fit. We calculate the average R^2 over 99 independent fittings for all 337 poses. Results show that there are 304 out of 337 fittings with R^2 greater than 0.99. In other words, in the most of pose examples, the manifold variation can be represented as a straight line in Eigenspace, and even in those cases where $R^2 < 0.99$, the derivation is small. Therefore, we feel justified in using this linear approximation in the following experiments.

To sum, for the two cases of the PDIS we considered, 1) where all pixels change with same linear function; 2) where each pixel has a different linear function, their projection forms a line in multidimensional Eigenspace. We proved this theoretically and showed examples. The images of one pose but different thermal states, as shown in Figure 5-7, form the second case of PDIS. The image set to generate object Eigenspace or universal Eigenspace in our thesis contains images of different poses, e.g., 337 poses in infrared image set, and is not PDIS but the combination of PDIS. The projection of the images from the same pose but different thermal states are not strictly in a line but can be approximated as a line.

5.3.2 Joint Effect of Thermal Variation- multi-part variation

In this section, we discuss multi-part variation, which is a joint effect of single-part variation. We consider three sets of training images: Landrover -- TSig2, TSig3 and

TSig4. The thermal variance accounted for each set is shown in the Figure 5-13. In each TSig, there are 10 different thermal states (T1, T2, ... T10): in TSig3, the thermal states (T1 to T10) varies in Engine part; in TSig4, the thermal states varies in Grill part; in TSig2, those varies in both Engine and Grill parts. In this case, TSig 4 and TSig 3 are two single- part variations, which we have discussed in last section. TSig 2 is the joint effect of them, the multi-part variation. The 10 images of 10 different thermal states and one pose form the second case of PDIS we discussed last section. Thus, they form a line in the Eigenspace. In this section, we discuss how to predict the position of the line formed by images from TSig2, given the images of TSig3 and TSig4.

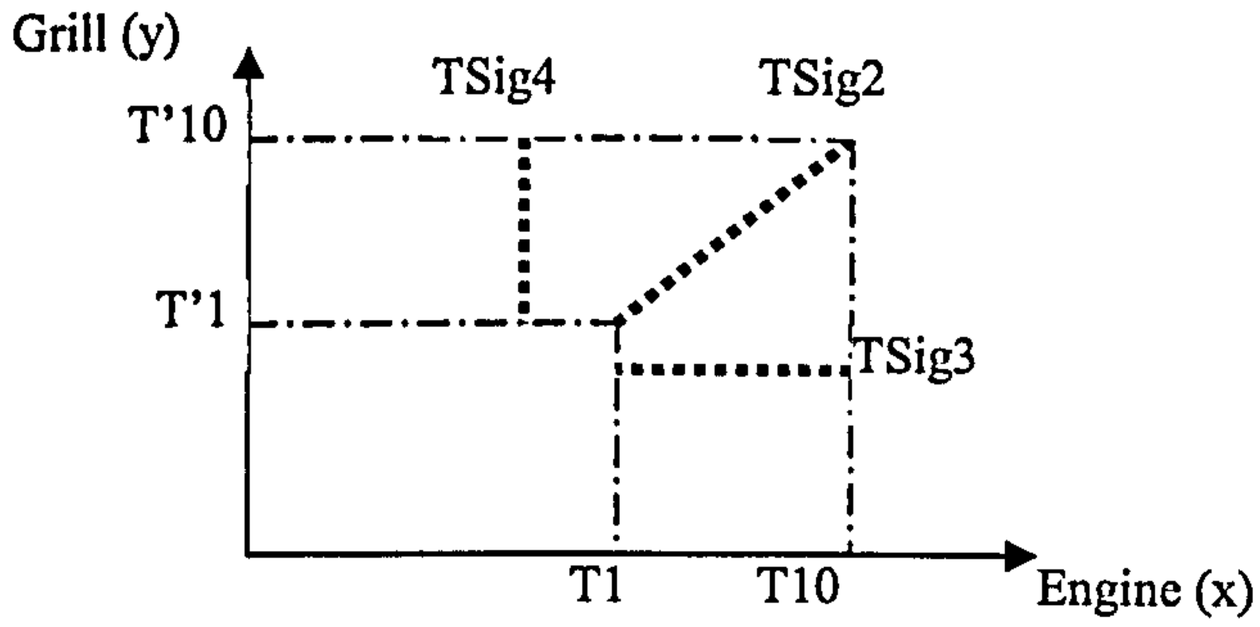


Figure 5-13 Landrover Tsig2, Tsig3 and Tsig4

Now we use the procedure in section 5.3.1 to find the rule of Eigenspace projections of these three image sets. We use all the 30 images to build the Eigenspace. The third 10 are images having joint effect of engine and grill thermal variation, the first and the second having engine part and grill part thermal variation respectively.

Original Image: $$\vec{i}_{x,y} = \{a_{co}, a_{ch1}, a_{ch2}\}, \quad (5\text{-}9)$$

where a_{ch1} and a_{ch2} are the two areas with intensity changing, referring to the example, they are engine part and grill part. a_{co} represents the rest area in the image which doesn't change among the images.

For the first 10 images: $a_{ch1} = f_k(x)$, $a_{ch2} = a_{c2}$, where a_{c2} is a constant number;

For the second 10 images: $a_{ch1} = a_{c1}$, $a_{ch2} = g_k(y)$, where a_{c1} is a constant number;

For the third 10 images: $a_{ch1} = f_k(x),\ a_{ch2} = g_k(y)$.

In the above definitions, x represents different thermal states in engine and y represents those in grill, as shown in Figure 5-13. For the first 10 images, $x = T1, T2, \cdots T10$; for the second 10 images, $y = T'1, T'2, \cdots T'10$; for the third 10 images, $(x, y) = (T1, T'1), (T2, T'2), \cdots (T10, T'10)$. Note that in the example 30 image data sets, the third 10 images are just one type of combination of the two PDIS. However, the theory presented here are general in that it counts all possible combinations of the two PDIS, e.g., the (x, y) could be $(Ta, T'b)$ where a and b are not equal.

Mean Image:
$$\bar{\bar{i}} = \sum_{x=1,y=1}^{n} \bar{i}_{x,y} \Big/ n \qquad (5\text{-}10)$$

$$\forall k \in a_{co} \quad \bar{\bar{i}}(k) = \bar{i}_{x,y}(k), \quad x \in \{1,2\cdots n\}, y \in \{1,2\cdots n\}$$

$$\forall k \in a_{ch1} \quad \bar{\bar{i}}(k) = \frac{2}{3}\sum_{x=1}^{10} f_k(x) + \frac{1}{3}a_{c1}$$

$$\forall k \in a_{ch2} \quad \bar{\bar{i}}(k) = \frac{2}{3}\sum_{y=1}^{10} g_k(y) + \frac{1}{3}a_{c2}$$

Differential Images:
$$\bar{i}_{D(x,y)} = \bar{i}_{x,y} - \bar{\bar{i}} \qquad (5\text{-}11)$$

$$\forall k \in a_{co} \quad \bar{\bar{i}}_{D(x,y)}(k) = 0$$

$$\forall k \in a_{ch1} \quad \bar{\bar{i}}_{D(x,y)}(k) = \bar{i}_{x,y} - \frac{2}{3}\sum_{x=1}^{10} f_k(x) - \frac{1}{3}a_{c1}$$

$$\forall k \in a_{ch2} \quad \bar{\bar{i}}_{D(x,y)}(k) = \bar{i}_{x,y} - \frac{2}{3}\sum_{y=1}^{10} g_k(y) - \frac{1}{3}a_{c2}$$

When building the Eigenspace, we find that any elements in eigenvectors corresponding to constant area in the image are zero (see Equation (5-5)). Then we project the image vectors into the Eigenspace, the coefficients are calculated by:

For engine part thermal variation:

$$c_j(x,y) = \left(\sum_{k \in a_{ch1}} \left(f_k(x) - \frac{2}{3}\sum_{x=1}^{10} f_k(x) - \frac{1}{3}a_{c1} \right) + \sum_{k \in a_{ch2}} \left(\frac{2}{3}a_{c2} - \frac{2}{3}\sum_{y=1}^{10} g_k(y) \right) \right) \cdot \bar{e}_j(k) \qquad (5\text{-}12)$$

The distance between two nearest points in engine part variation in j th dimension is

$$\bar{e}_j(k) \cdot \sum_{k \in a_{ch1}} \left(f_k(x+1) - f_k(x) \right)$$

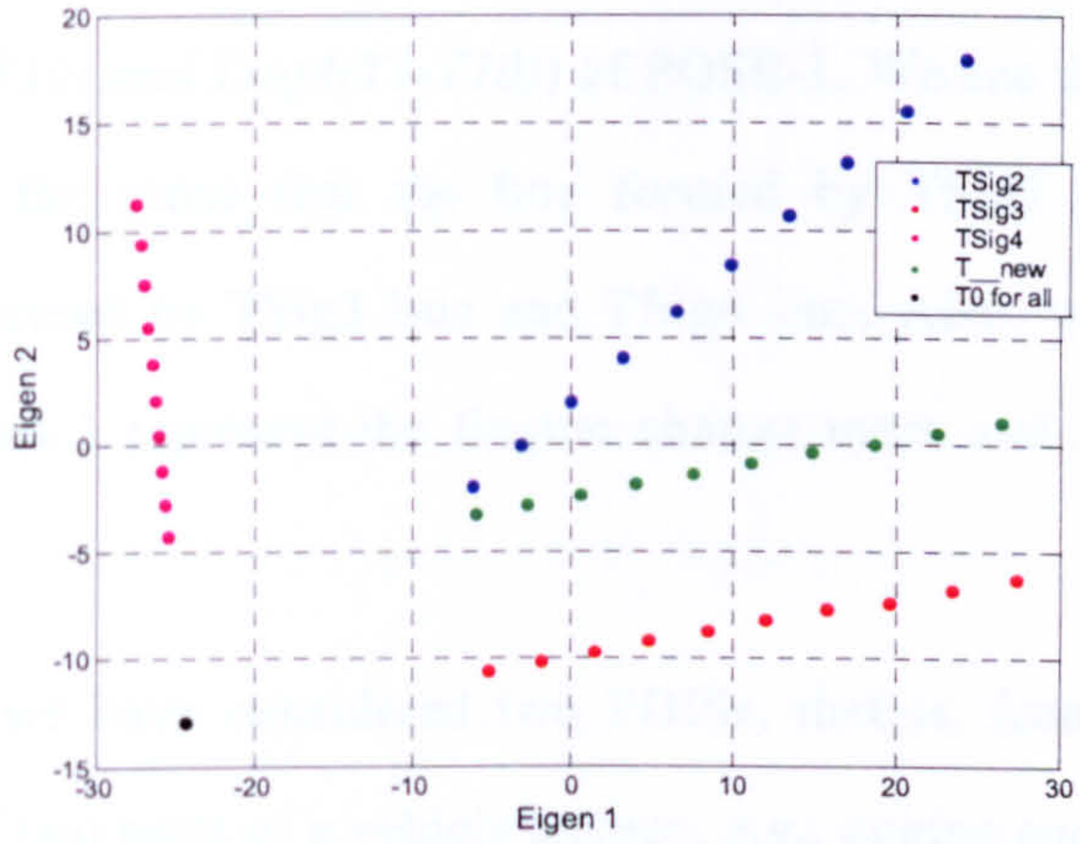


Figure 5-14 TSig3 (Engine), TSig4 (Grill) and TSig2 (Engine and Grill) in Landrover dataset

For grill part thermal variation:

$$c_j(x,y)=\left(\sum_{k\in a_{ch1}}\left(-\frac{2}{3}\sum_{x=1}^{10}f_k(x)+\frac{2}{3}a_{c1}\right)+\sum_{k\in a_{ch2}}\left(g_k(y)-\frac{1}{3}a_{c2}-\frac{2}{3}\sum_{y=1}^{10}g_k(y)\right)\right)\cdot\vec{e}_j(k) \quad (5\text{-}13)$$

The distance between two nearest points of grill part variation in j th dimension is

$$\vec{e}_j(k)\cdot\sum_{k\in a_{ch2}}\left(g_k(y+1)-g_k(y)\right)$$

For joint effect of the two parts:

$$c_j(x,y)=\left(\sum_{k\in a_{ch1}}\left(f_k(x)-\frac{2}{3}\sum_{x=1}^{10}f_k(x)-\frac{1}{3}a_{c1}\right)+\sum_{k\in a_{ch2}}\left(g_k(y)-\frac{1}{3}a_{c2}-\frac{2}{3}\sum_{y=1}^{10}g_k(y)\right)\right)\cdot\vec{e}_j(k) \quad (5\text{-}14)$$

The distance between two nearest points of joint effect in j th dimension is

$$\vec{e}_j(k)\cdot\left(\sum_{k\in a_{ch1}}\left(f_k(x+1)-f_k(x)\right)+\sum_{k\in a_{ch2}}\left(g_k(y+1)-g_k(y)\right)\right)$$

From the above analysis, we see that in each dimension in Eigenspace, the distance of the nearest two points of the joint thermal variation is the sum of the distance of the points in two single thermal variations. Thus, the line formed by the joint variation is the diagonal of the parallelogram formed by two lines of two single variation in multidimensional space.

Figure 5-14 show the Eigenspace formed by images from the 30 thermal states (*Tsig2(T1-T10), Tsig3(T1-T10) and Tsig4(T1-T10)*) of POSE-1. We see that Figure 5-14 is similar as Figure 5-13 in the sense that the line formed by TSig2 is on the diagonal of the parallelogram formed by TSig3 line and TSig4 line. Also, we see that in this case, the direction of Eigen 1 represent the Engine change more and Eigen 2 represent the Grill change more.

In this section, we have considered two PDISs, that is, from a base thermal state, the thermal states of two parts of a vehicle change, e.g., engine and grill. We develop a theory to predict the Eigenspace projection of the combination of those two thermal states changing given the two individual Eigenspace projections. The theory is general in that it considers all possible types of combinations although the example used in this section represents only one type. The theory can also be used for prediction of a combination of more than two single thermal states changing. If we take 3 thermal states for example, we can predict the combination of the first 2 and then predict the combination of its result and the 3rd.

5.3.3 Proposed Algorithm

In this section, we propose an algorithm specially designed for thermal imagery based on the deformable manifold. We use examples to demonstrate why this algorithm is more advanced than the traditional one. The idea of a 'Deformable' manifold is that instead of using images from all thermal states, we train on only a few thermal state and the recorded 'directions'.

In the last two sections, we proved that the thermal images having single part variation and multipart variation form lines in the Eigenspace. Based on this fact, we predict that any other images with the same type of thermal variation but in different scales and not included in the training set will also fall in to the line of direction for each pose when projected to Eigenspace.

We start from an example image set and analyze the deformable method in comparison with original Eigenspace based object recognition. The image set we choose contains images of a Landrover-Freelander with 337 poses and 10 thermal states: T1, T2, ... T10. The ten thermal states are formed by thermal variations in the whole body of the vehicle. Figure 5-15 shows the appearance of different thermal states using pose 66 of the imaged object. Figure 5-16 shows the Eigenspace of the image set.

We use the images of the first 3 thermal states as the training set and images from other 7 thermal states as test images. The question is -- can the pose still be estimated correctly if the thermal variation goes beyond the training set?

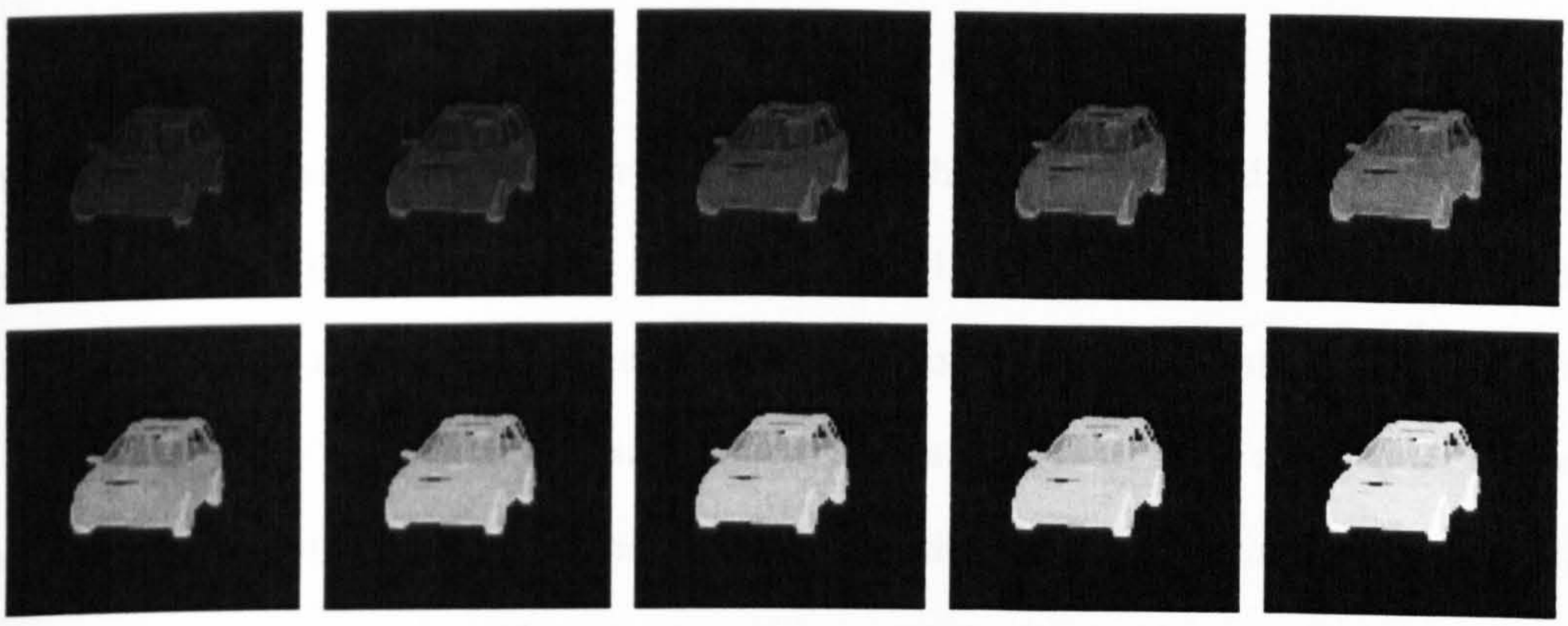


Figure 5-15 Landrover - Freelander (pose 66 out of total 337 poses) of the 10 thermal states

The recognition process in the original Eigenspace based method is to find a nearest neighbour of the projection of the test image in the Eigenspace points of the training image. Here we analyze one test for example, test image of (***Pose 2, T10***). In the nearest neighbour method, it's recognized as pose 189. We also find that the nearest training image to *pose2 is* the 39th neighbour of (*Pose2, T10)*. However, we find that the distance between (*Pose 2, T10*) and the line formed by the training images of Pose 2, (T1, T2, T3) is smaller than the distance between (*Pose 2, T10*) and training images of Pose 189. This implies that to find the nearest line can be more accurate than to find the nearest neighbour.

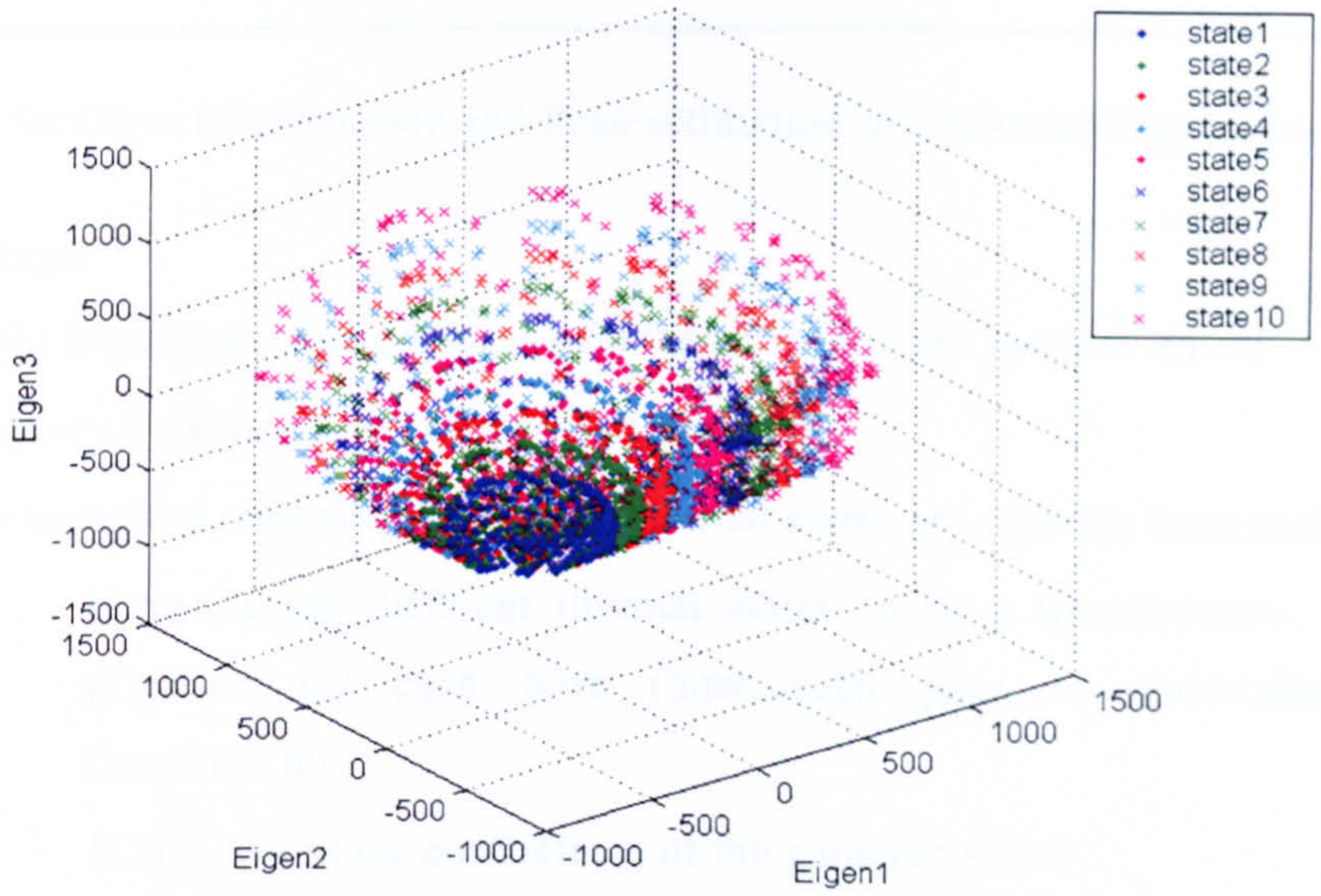


Figure 5-16 Eigenspace of T1-T20

Table 5-1 shows the proposed algorithm. Because the classifier used in the recognition process is to find the nearest line, hereafter we call this algorithm the **NL** algorithm. Similarly, we call the original Eigenspace based method the **NN** algorithm, short for nearest neighbour. For the simulated infrared datasets, there are some poses whose appearance does not change along the thermal state change, e.g., the thermal state change in the engine part does not affect the appearance of the back of a car. Thus, instead of forming a line, those image projections are gathered in one point. Considering of this problem, we adjust the NL approach to a conditional solution (see step 3 in training stage and step 2 in recognition stage).

The algorithm in Table 5-1 is the algorithm for object recognition in universal Eigenspace. As stated in Section 3.1.5, the universal Eigenspace is built by projecting images of all the training objects. In universal Eigenspace, object recognition and pose estimation are done in the same step. When considering object Eigenspaces, identifying object and identifying pose can be done in separate steps: first, find the correct object subspace; then find the correct pose in that subspace. In the object Eigenspace based method, the algorithm in Table 5-1 can be used as a pose estimation step. The probabilistic method described in section 3.5 is used as an object identification step.

Algorithm for Object Recognition and Pose estimation in Universal Eigenspace:

Training Stage:

1. Build Eigenspace using images of different poses and thermal states;
2. Project all training sets to Eigenspace;
3. For each pose, examine if different thermal states are separate from each other
   - If yes, using different thermal states to fit a line/direction, record the direction for each pose (now each pose is represented by the line/direction);
   - If not, record the coefficients of the gathered point.

Recognition Stage:

1. Project the input image to Eigenspace;
2. Compute the distance between the input image position to the representation of different poses:
   - If the pose is recorded as a line, compare the distance between the input point and the line;
   - If the pose is recorded as a point, Compare the distance between the input point and the point;
3. Identify the direction associate with the shortest direction as the estimated pose;

Table 5-1 Proposed NL Algorithm

In the proposed algorithm, there are two differences from traditional Eigenspace method:

1. multidimensional line fitting in step 3 of the training stage;
2. calculating distance between a point and a line in multidimensional space in step 2 in the recognition stage.

Line fitting and calculating the distance between a point and a line are quite standard. However, the solution to those two problems in multi-dimensional space rarely appears in the literature. Appendix B shows the details of the calculation.

5.4 Experiments

The experiments in this section are to test the proposed algorithm in section 5.3.3 which deals with thermal variation in pose estimation and object identification and compare the results from the new proposed algorithm with the results from the traditional method. We also test our theory of the joint effect of thermal variation in subspace stated in section 5.3.2, and compare the result of the predicted training set with the real training set using both new and traditional methods. Finally, we test several large scenes which contain one or more objects with different thermal states, at different scale, and with clutter and occlusion.

The best way to test the method is to use real images for both the training set and the test image, as in Nayar's work [69]. In their work, training images of toy objects were obtained by placing an object on a turntable and varying the degree of rotation (θ_1 in Figure 2-9). However, in our case, the objects of interest are vehicles, the viewpoints designed are the 337 vertices of a upper sphere of 3rd level Icosahedron, and the thermal variation is considered. It is difficult to get real images of such a training image set. Therefore, for training set, we use simulated IR images generated by CAMEO-SIM which has been assessed and proved to be a proper IR image simulation package, see section 5.2. Because the real vehicles of vehicle models we used for training set is hard to get, for test images, we still use simulated images. For the large infrared scenes in section 5.4.4, we use real infrared scene and stick our simulated vehicles in it to demonstrate the whole recognition process starting from the segmentation.

5.4.1 Experiments on whole body thermal variation

Objective:

This experiment is to compare the NN and NL approach using relatively simple image sets: an image set of Landrover-freelanders and another of Panthers with different thermal states. We have described and shown example images of the Landrover-freelanders image

set in section 5.3.3 and in Figure 5-15. Figure 5-18 shows the example images from different thermal states of the Panther. The thermal variation of each pose in these image sets belongs to the first situation described in section 5.3.1, that is, the sensed thermal radiation of the whole vehicle changes following one linear equation. In addition, we examine the NL method on the images with noises.

Procedure:

The training set and test images are the same as in the simulation in section 5.3.3: we train the first 3 thermal states and test using the other 7 thermal states to estimate the correct pose. We compare the pose estimation results of the NN and NL methods using 100 eigenvectors for each method. There is no energy normalization of the images in both training and recognition process. In testing with noisy images, a Gaussian distribution $N(\mu, \sigma^2)$ with a mean (μ) of '0' and variance (σ^2) of 0.1 is added to each test image (see 3.6.2).

Result and Discussion:

Figure 5-17 shows the results of the NN and NL methods. We see a better performance of NL approach over NN approach in term of recognition results for both the landrover-freelander image set and panther image set. In the Landrover-freelander image set, using NN method, the recognition rate of the first thermal state is 100%. As the test thermal state moves away from the training thermal states, the recognition rate decreases. For the 7th thermal state, 87 out of a total 337 poses are recognized as other poses. For the Panther image set, the recognition rate of NN is even worse. For both image sets, using the NL method, the recognition rate for each thermal state is maintained at 100%.

Figure 5-20 shows the results of pose identification of images from thermal state 7 for all 11 objects. We see that for all the test objects, the NL performs much better than NN. From Figure 5-21, we see that the results of NL method are not affected very much when the test image is noised while NN method are.

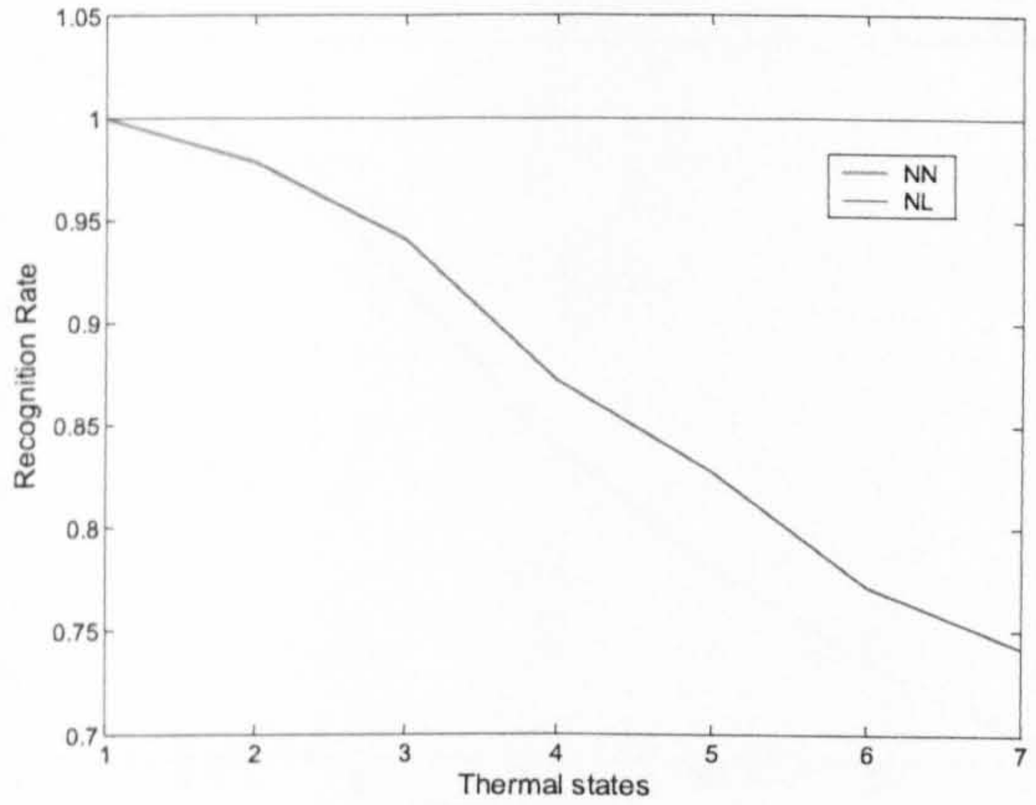


Figure 5-17 Comparison between *NN* approach and *NL* approach, Landrover-Freelander, No Noise

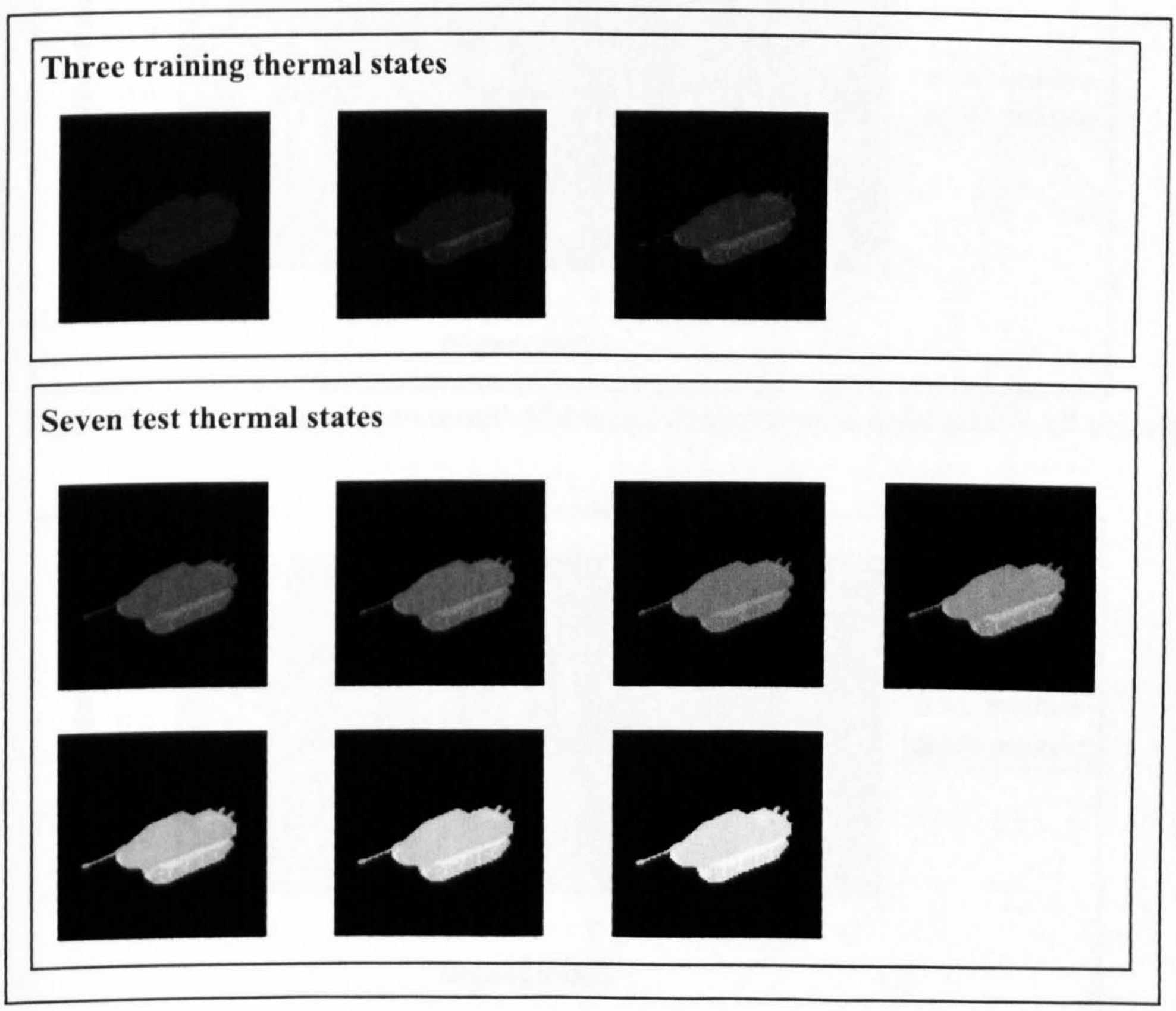


Figure 5-18 Example training and test images of Panther

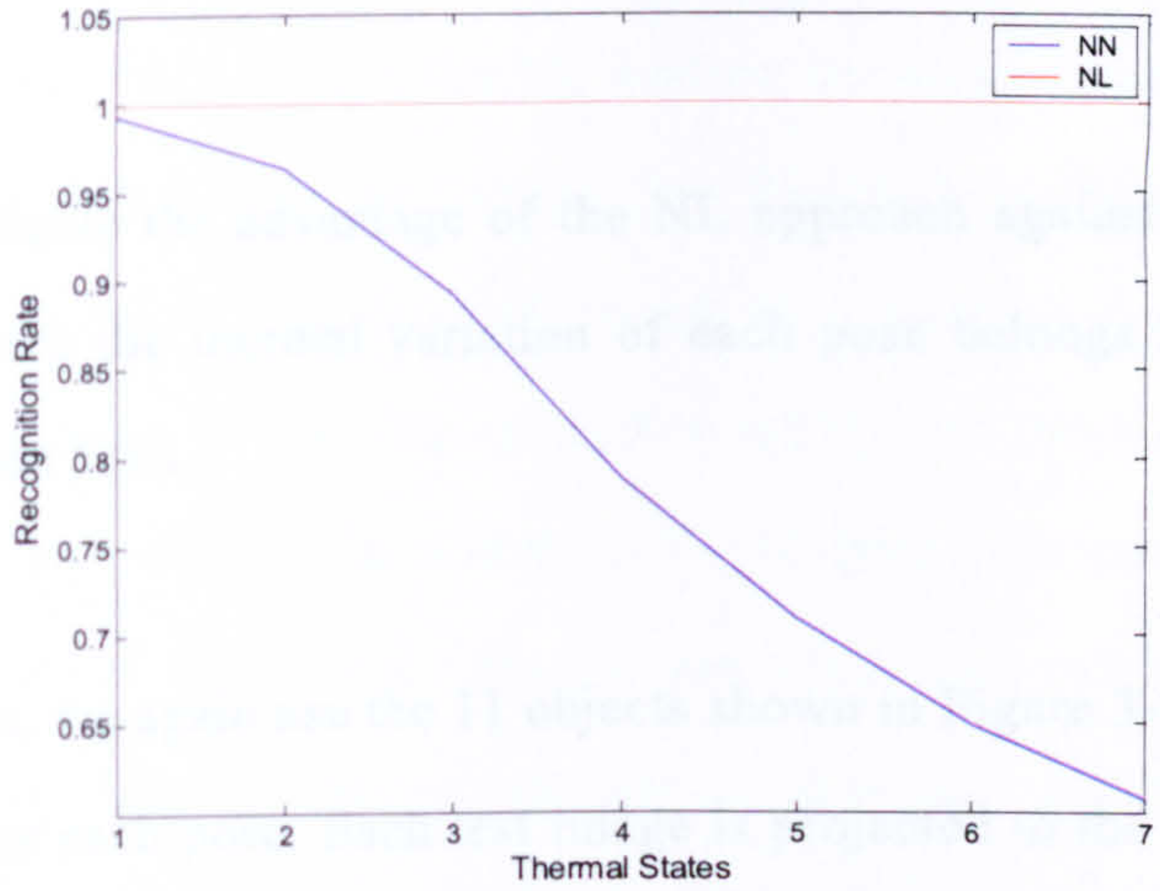


Figure 5-19 Comparison between *NN* approach and *NL* approach, Panther, No Noise

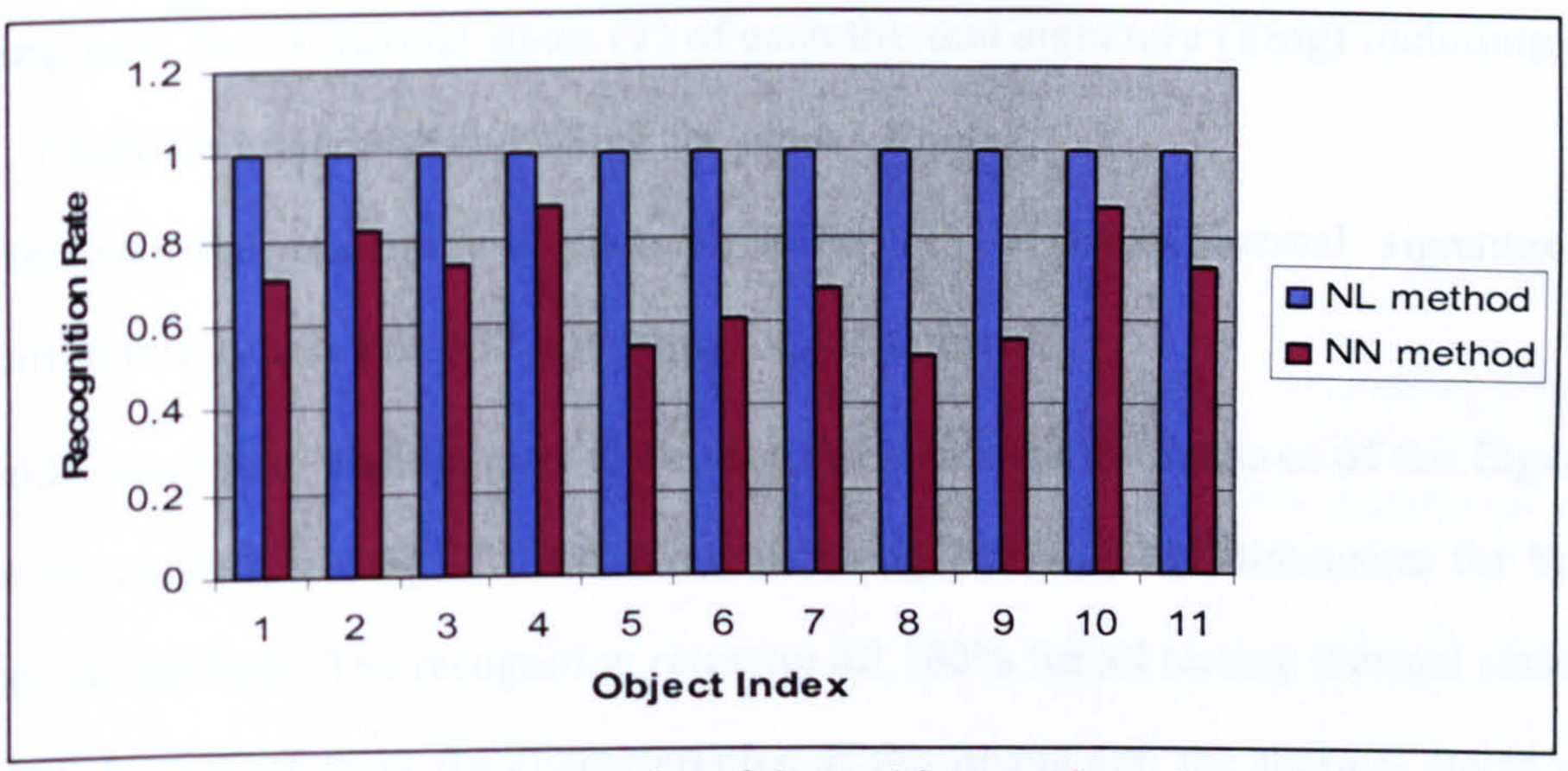


Figure 5-20 Recognition results of thermal images from thermal state 7, all objects

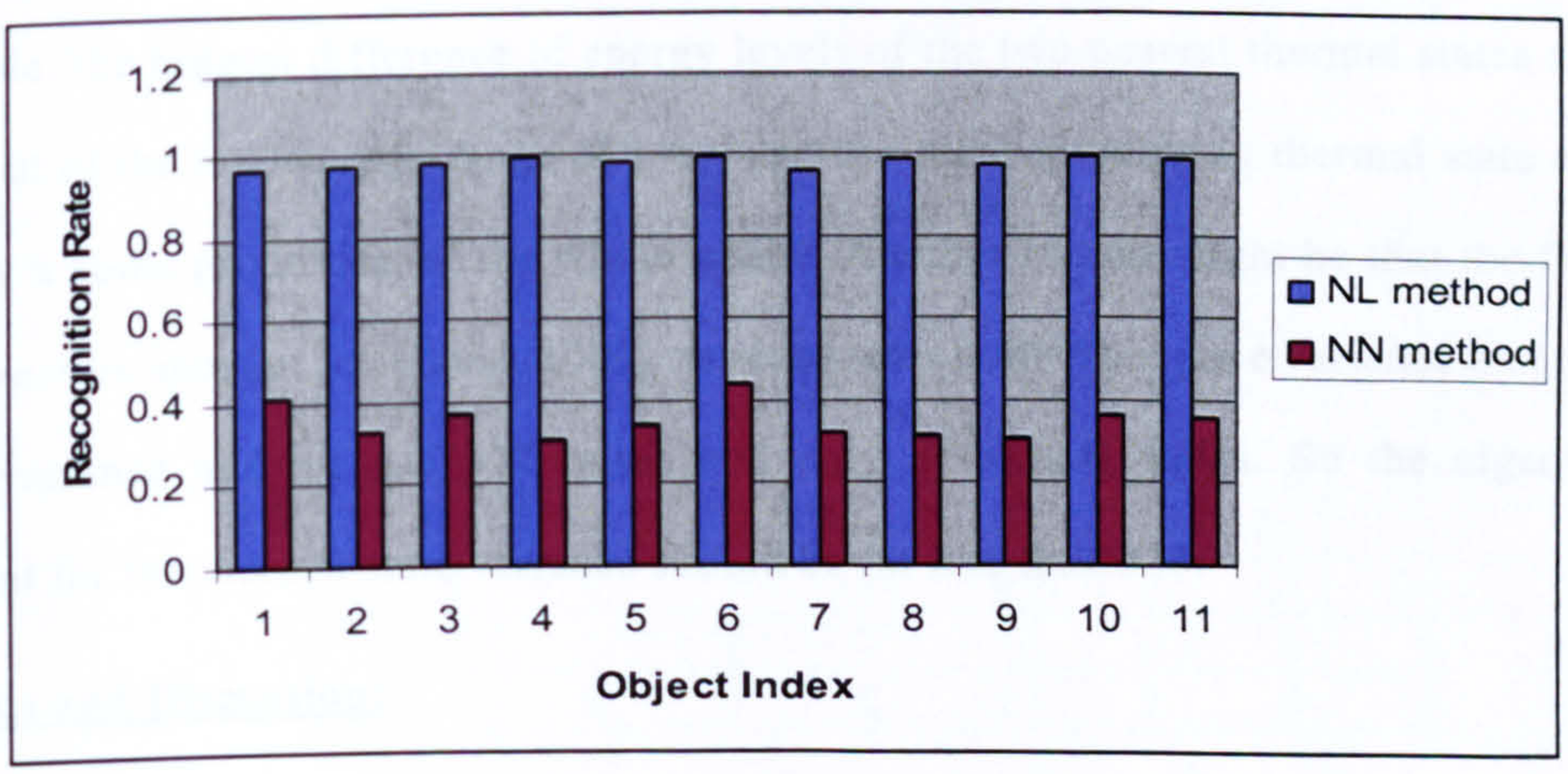


Figure 5-21 Recognition results of thermal images from thermal state 7, all objects, with Gaussian noise (0, 0.1).

5.4.2 Experiments local part thermal variation

Objective:

We aim to investigate the advantage of the NL approach against NN on more complex image sets in which the thermal variation of each pose belongs to the second situation described in section 5.3.1.

Procedure:

In this experiment, we again use the 11 objects shown in Figure 3-38. Object Eigenspaces are constructed for each pose. Each test image is projected to the 11 object Eigenspaces. The smallest out-of-space error is used to do object recognition and the NL method is used for pose identification.

Training sets: first 3 thermal states (T) of each thermal signature (TSig) including TSig2, TSig3, TSig4 for Landrover and TSig1 for other vehicles;

Test images: the remaining 7 thermal states (T) of each thermal signature, each recognition rate is based on 337 test poses for each object.

To decide how many eigenvectors to be used, we examine the features of the Eigenspace formed by Landrover Tsig3 T1-T3. We tried using 300 and 100 dimension for both the NN and NL methods. The recognition rates are all 100% for all testing thermal states. One reason of this comes from the characteristics of the image set: the thermal states are not properly separated and images from different thermal state does not differ very much. For example, the biggest difference of energy levels of the two nearest thermal states are only 14.4 out of the total energy range of 1-60; and the part representing thermal state changes is only a some proportion of the whole image. Another reason might be that the first 300 eigenvectors account for almost 100% variance which involves pose variance and thermal state variance and we use 337 poses and only 3 thermal states. So the eigenvectors account for the thermal state variance should be far less than 300.

Results and Discussion:

To compare the NN and NL method, we use the first 20 dimensional Eigenspace for pose estimation. Six objects (Helicopter, Freelander, M1A1 Tank, Panther Tank, Missile

launcher, T80 Tank) got 100% recognition rate for all the poses and all the testing thermal states. We get a better result using the NL method for object Landrover Tsig3 and Tsig2, Car_Shadow, Car_Sedon and Car_Mirage (see Figure 5-24, Figure 5-26, Figure 5-28, Figure 5-30, and Figure 5-32). The NL shows an advantage over NN especially when thermal state is far from the training states. We see that NL method always gives a 100% recognition rate. We could predict that when the thermal state goes even farther from the testing states, the result of NN will getting worse while the result of NL could still remain good. For the other 6 objects, the two methods can give the same result.

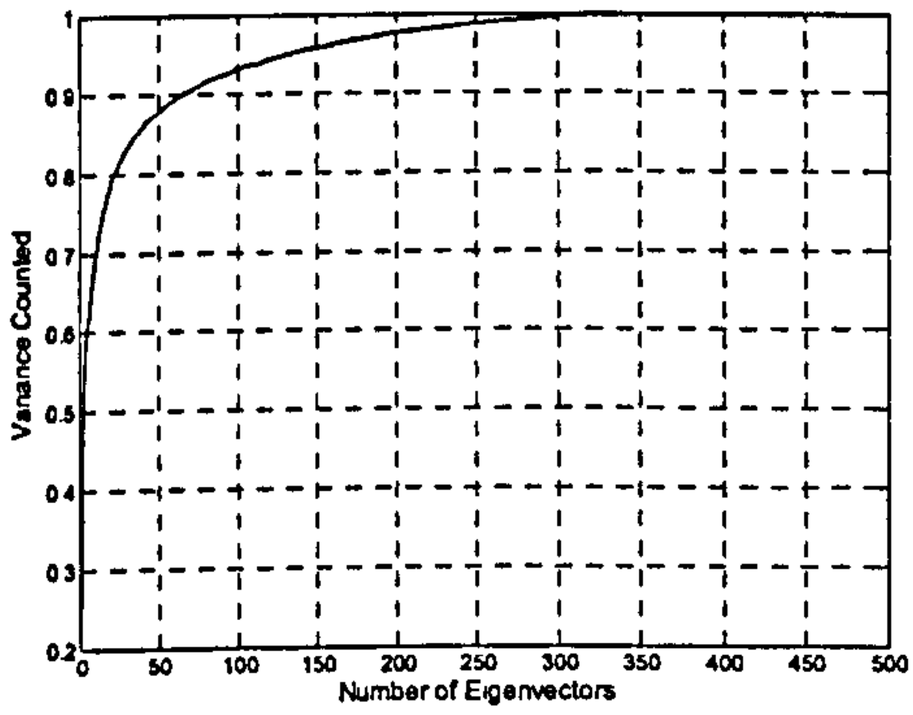


Figure 5-22 The variance accounted for by the first n Eigenvectors. (LandRover/Tsig3/T1-T3)E.g., the first 100 eigenvectors account more than 94% variance among the training images. And the first 300 eigenvectors account almost all the variances. This analyse is used to determine how many eigenvectors should be used in recognition stage.

The table below shows the object recognition result. The average object recognition rate is 95.6% using object Eigenspaces and 20 Eigenvectors.

Testing image / Recognize as	obj1	obj2	obj3	obj4	obj5	obj6	obj7	obj8	obj9	obj10	obj11
obj1	84	0	0	0	0	0	0	0	0	0	0
obj2	0	84	0	0	0	0	0	0	0	0	0
obj3	0	0	83	0	0	0	0	0	0	0	0
obj4	0	0	0	82	0	0	0	0	0	0	0
obj5	0	0	0	1	70	0	0	2	1	0	0
obj6	0	0	0	0	0	83	0	0	0	1	0
obj7	0	0	0	0	0	0	84	0	0	1	0
obj8	0	0	0	0	3	0	0	65	1	0	0
obj9	0	0	1	1	11	0	0	17	82	0	0
obj10	0	0	0	0	0	1	0	0	0	82	0
obj11	0	0	0	0	0	0	0	0	0	0	84
Recognition Rate (%)	100.0	100.0	98.8	97.6	83.3	98.8	100.0	77.4	97.6	97.6	100.0
Number of Eigenvectors: 20, Thermal state: 10, Object Eigenspaces											

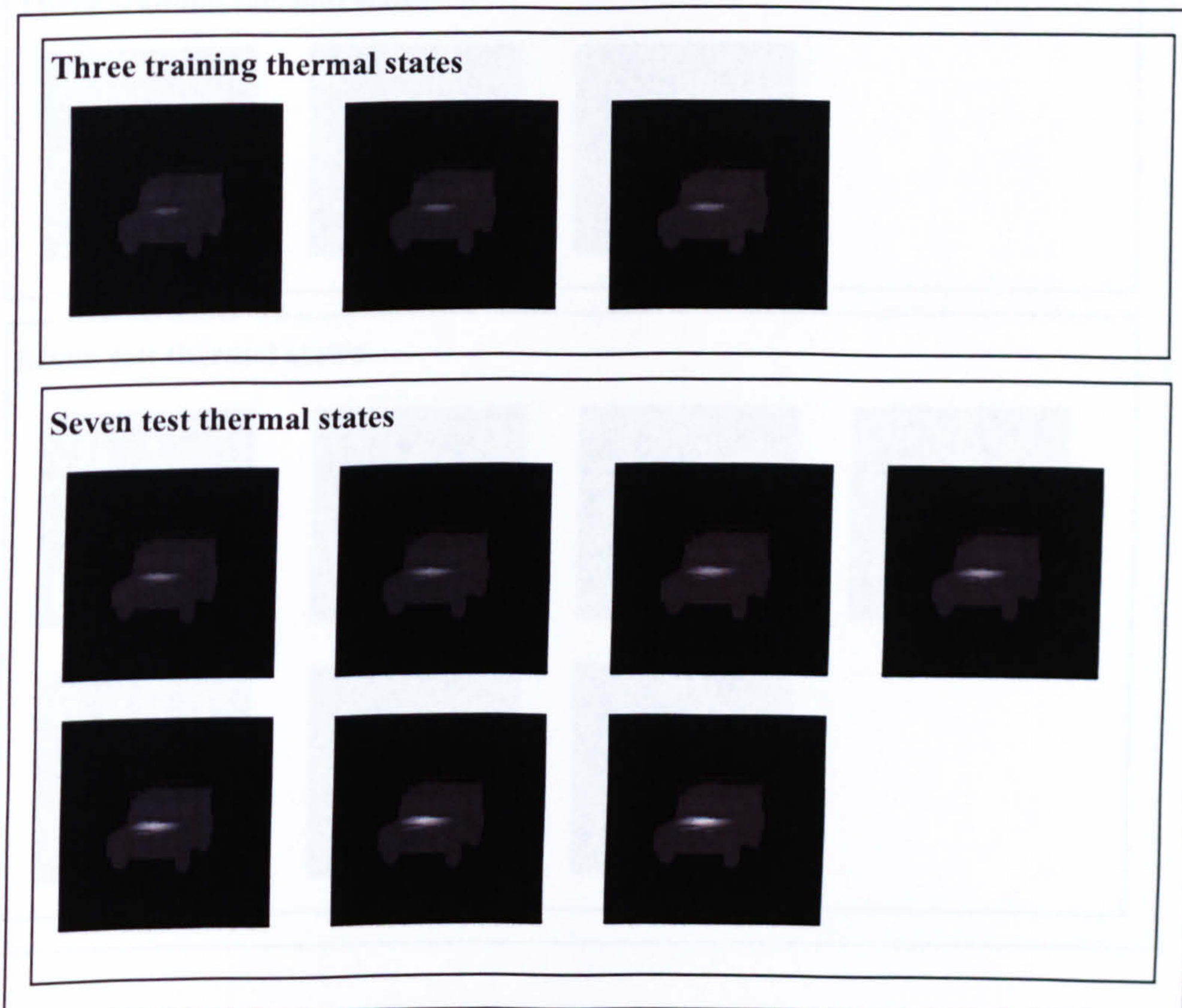


Figure 5-23 Examples of training and test images of Landrover TSig3

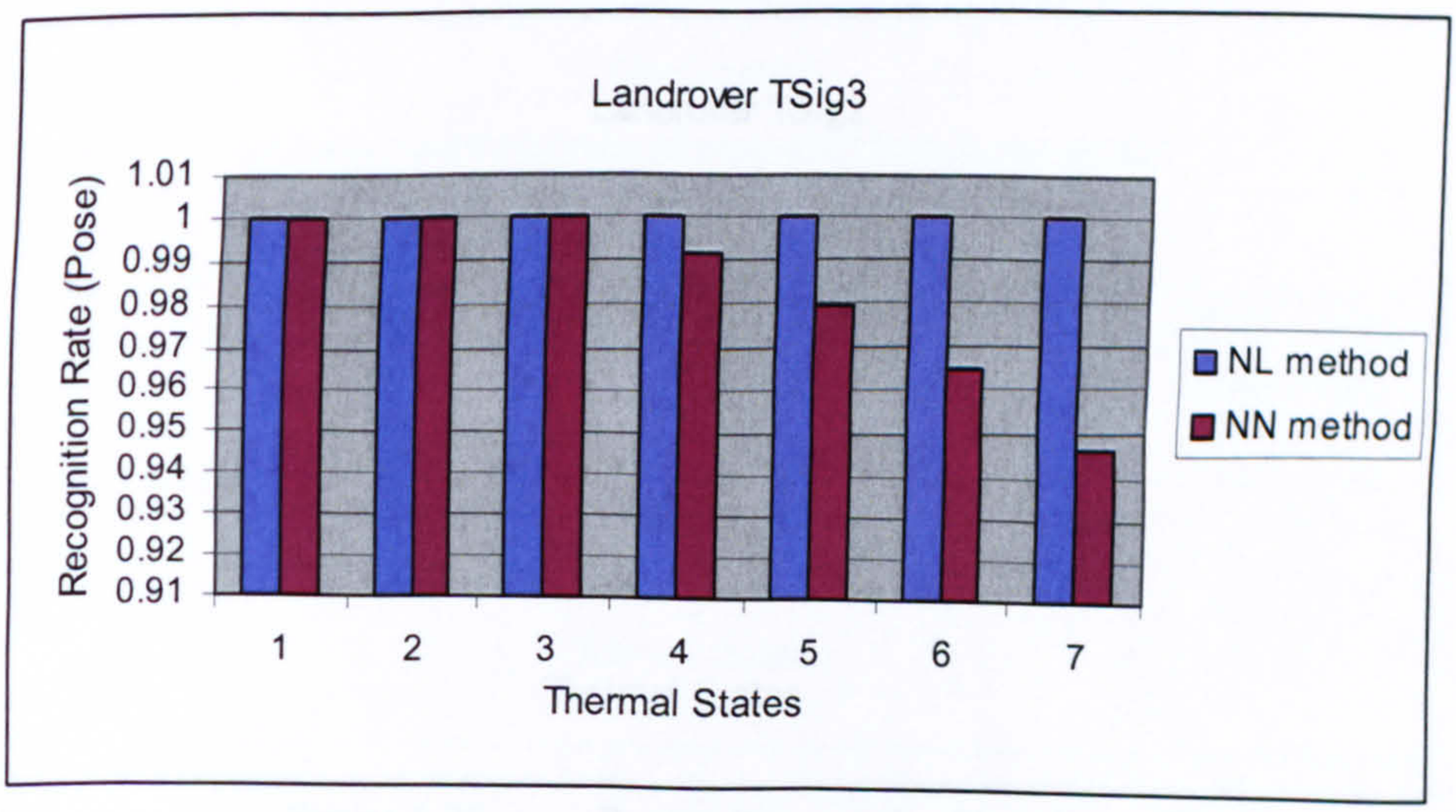


Figure 5-24 Recognition result of Landrover Tsig3

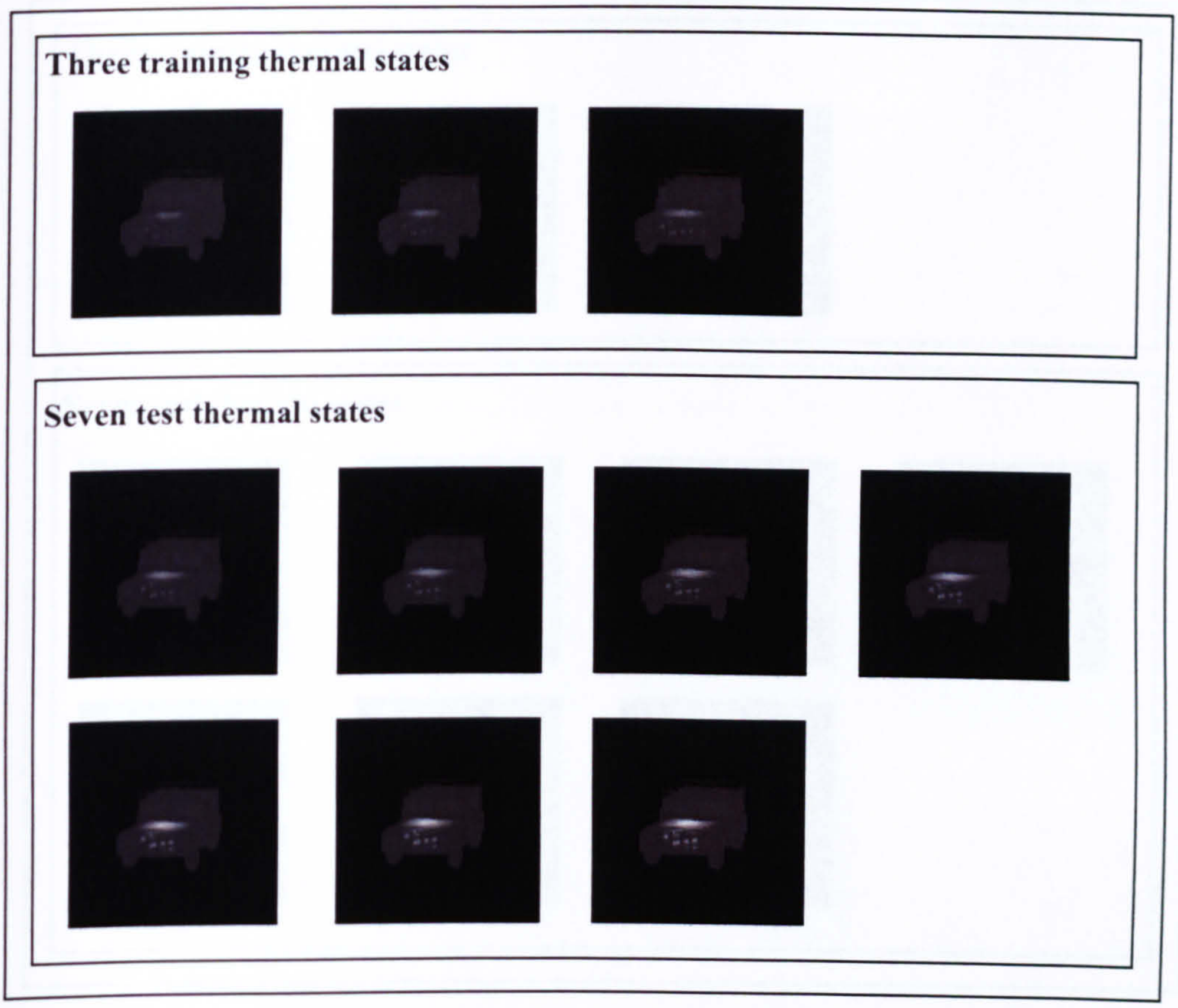


Figure 5-25 Examples of training and test images of Landrover TSig2

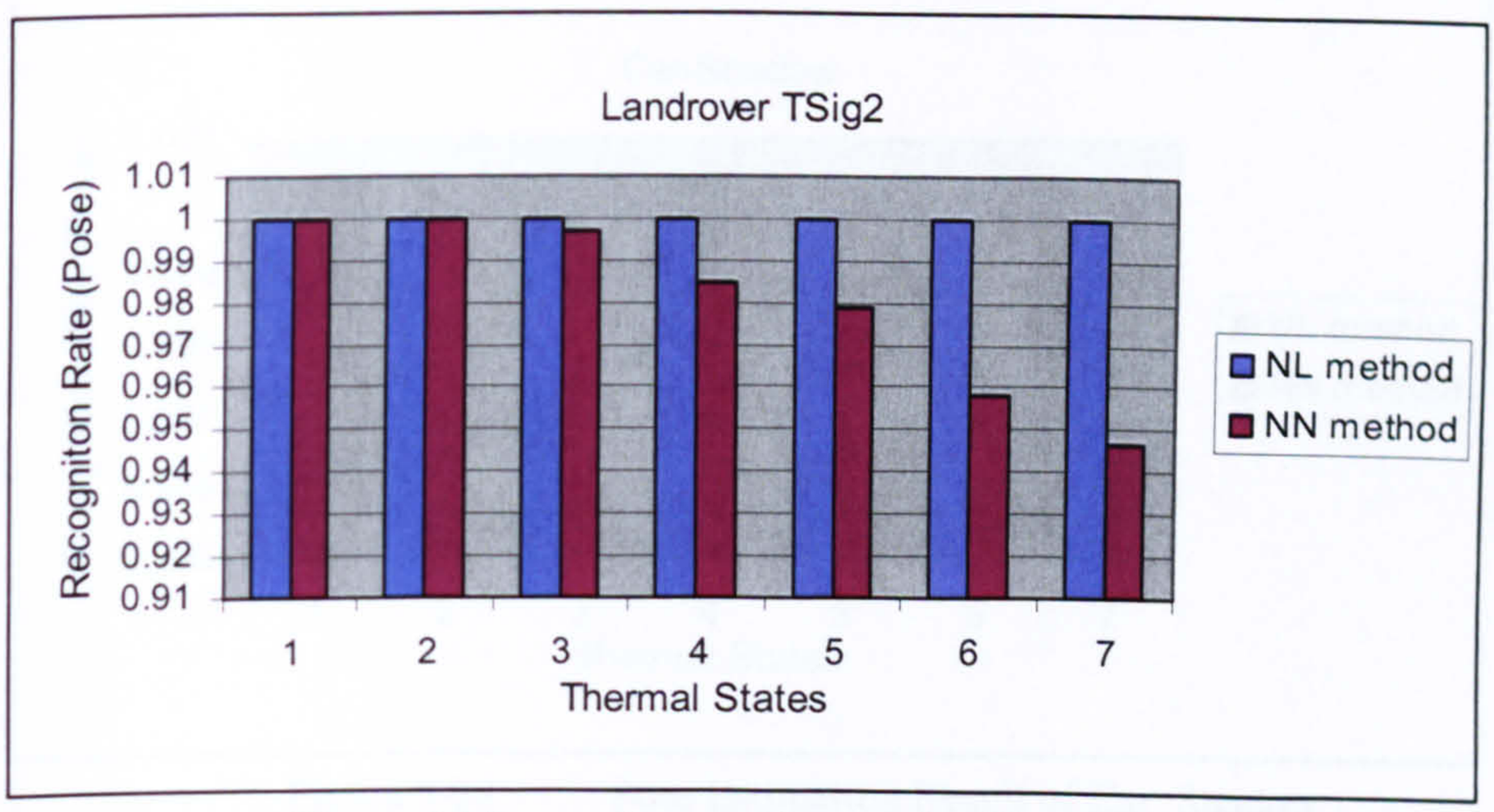


Figure 5-26 Recognition result of Landrover Tsig2

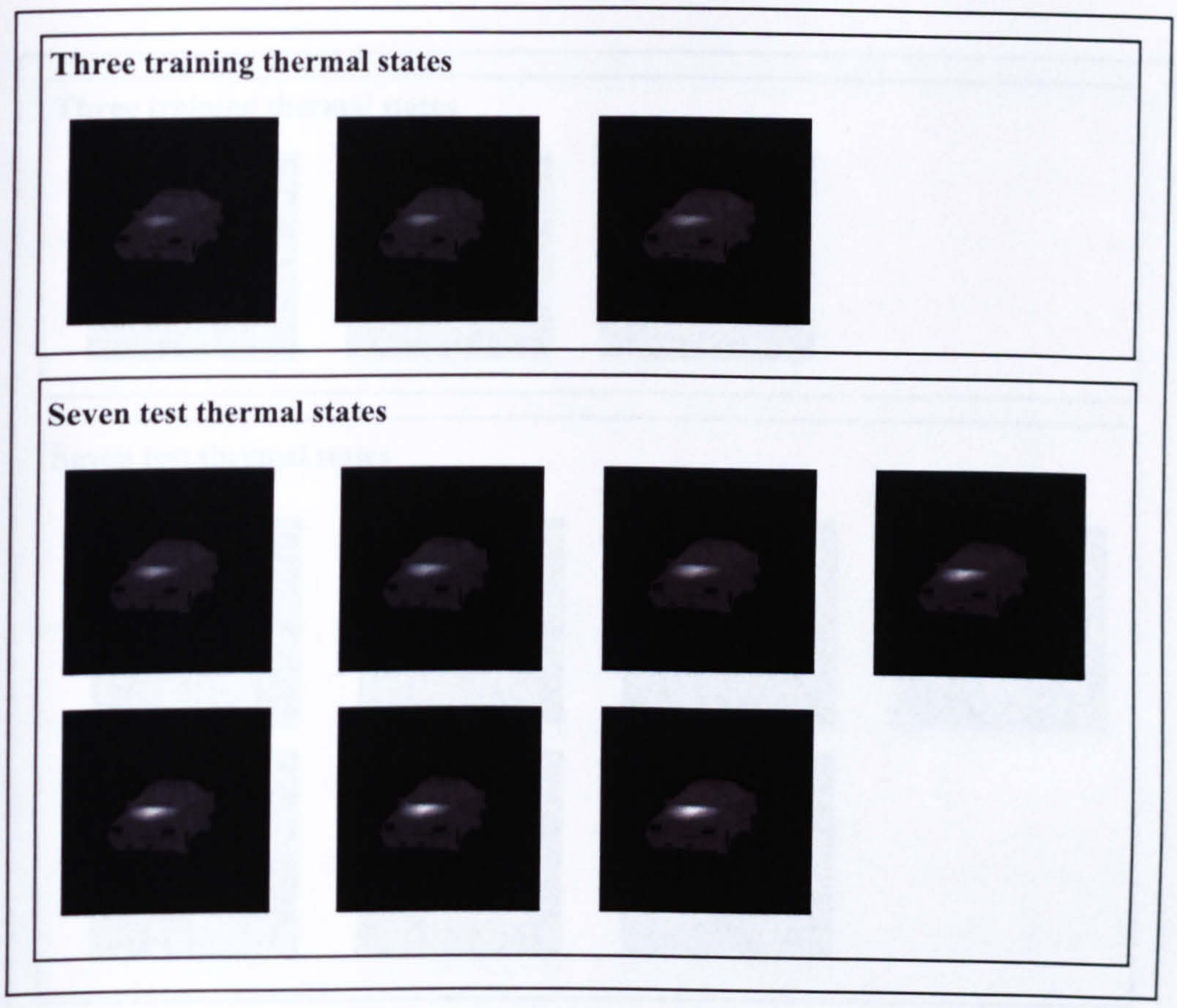


Figure 5-27 Examples of training and test images of Car-Shadow

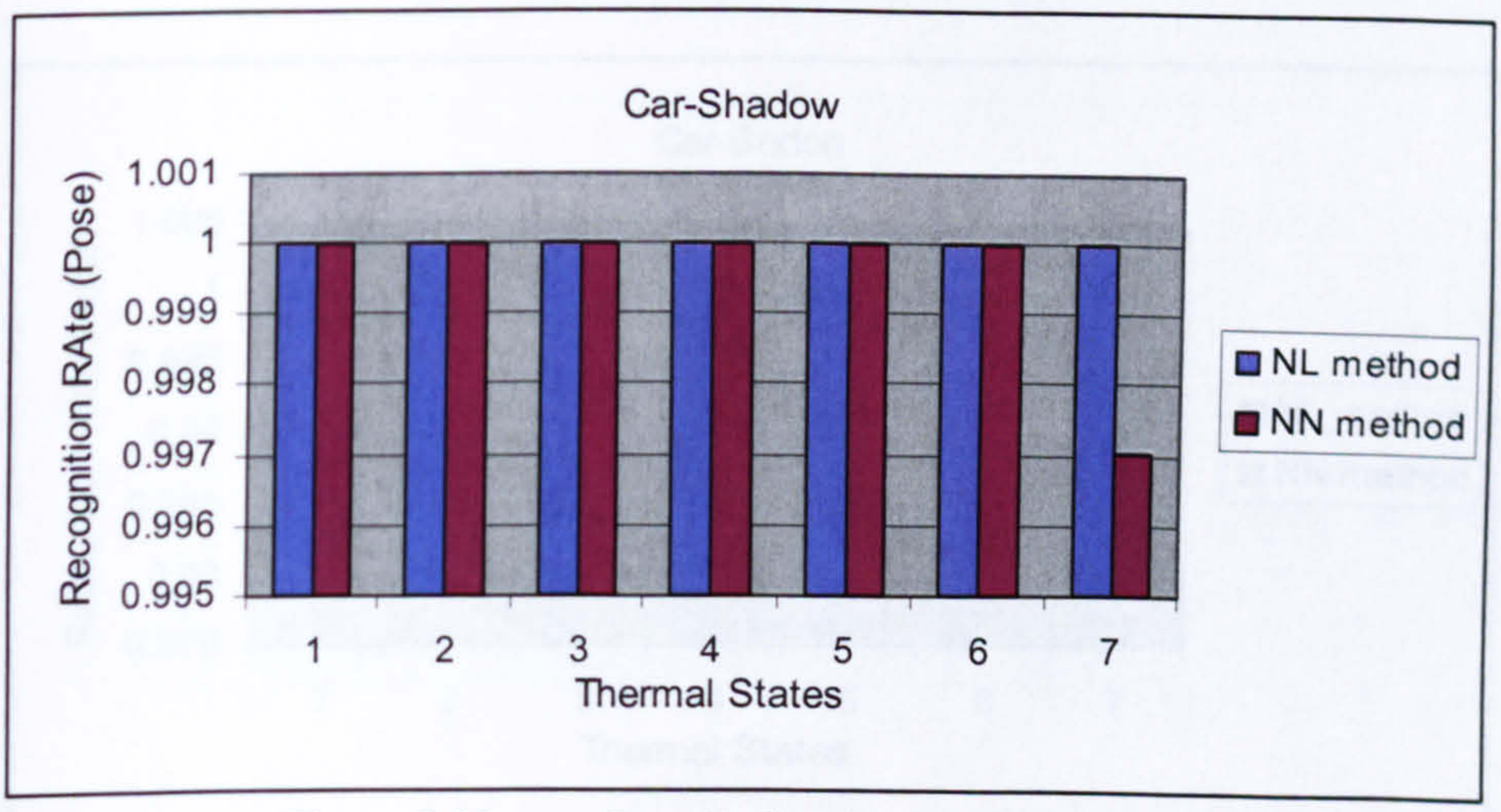


Figure 5-28 Pose Estimation Result of Car_Shadow

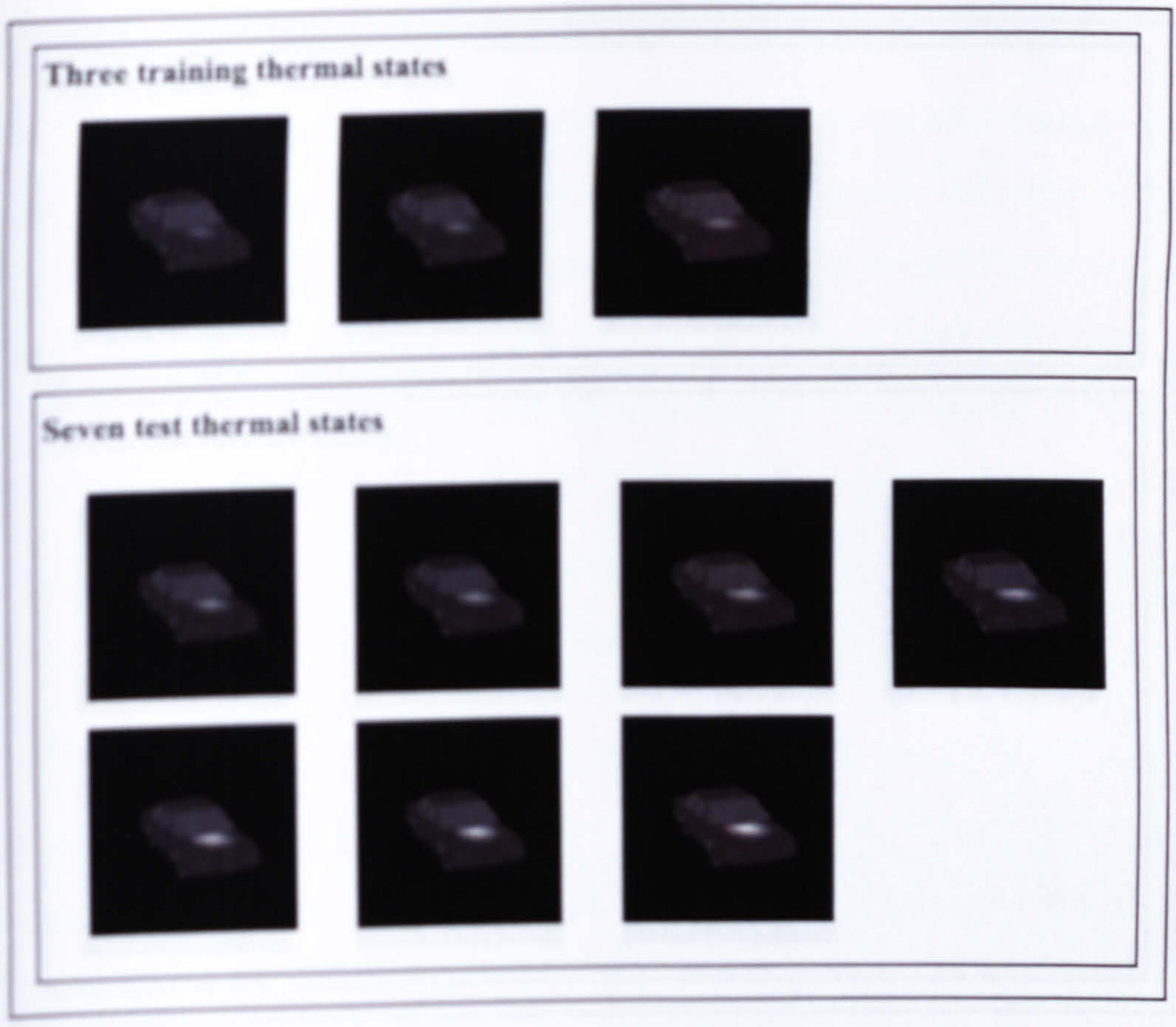


Figure 5-29 Examples of training and test images of Car-Sedon

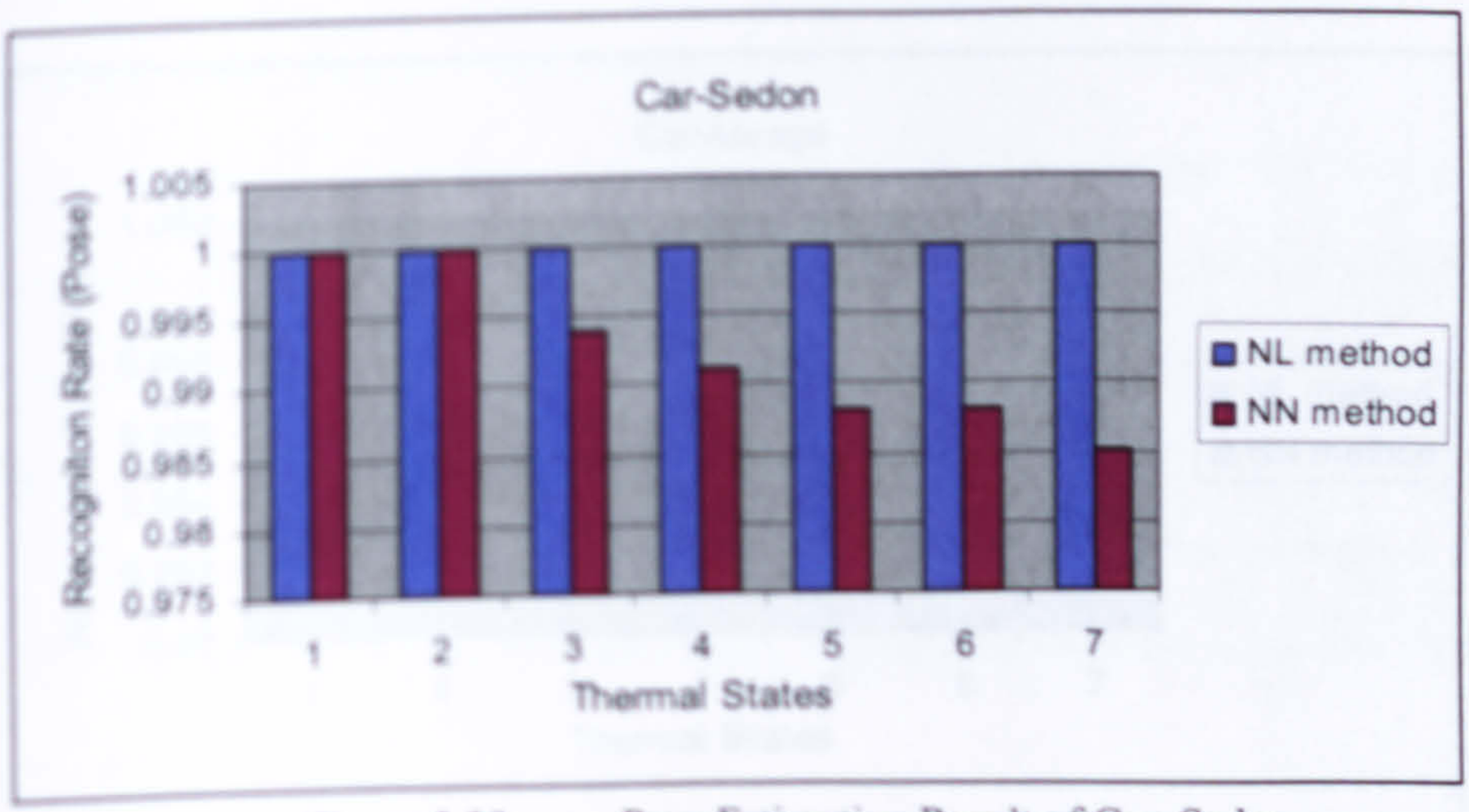


Figure 5-30 Pose Estimation Result of Car_Sedon

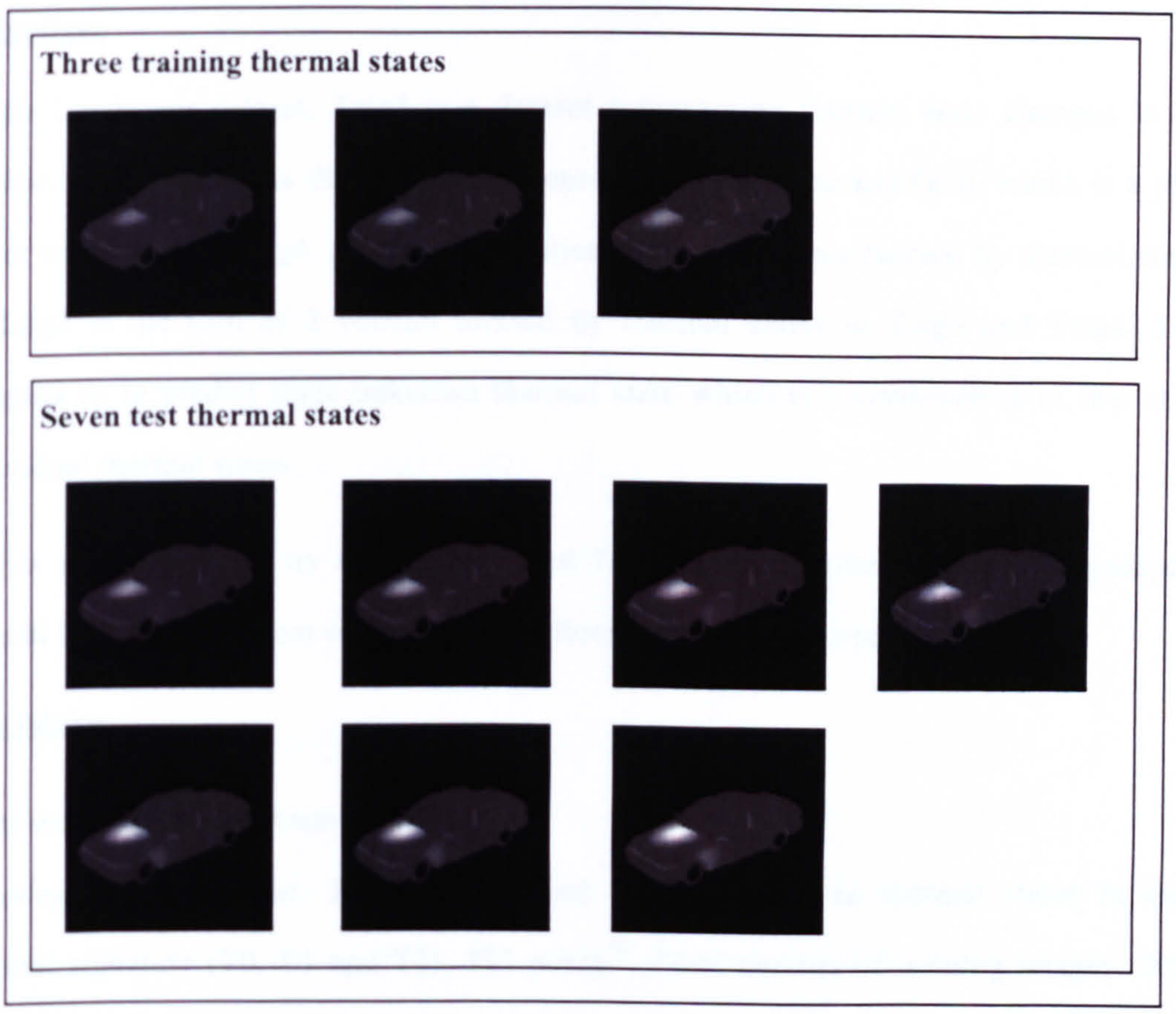


Figure 5-31 Examples of training and test images of Car-Mirage

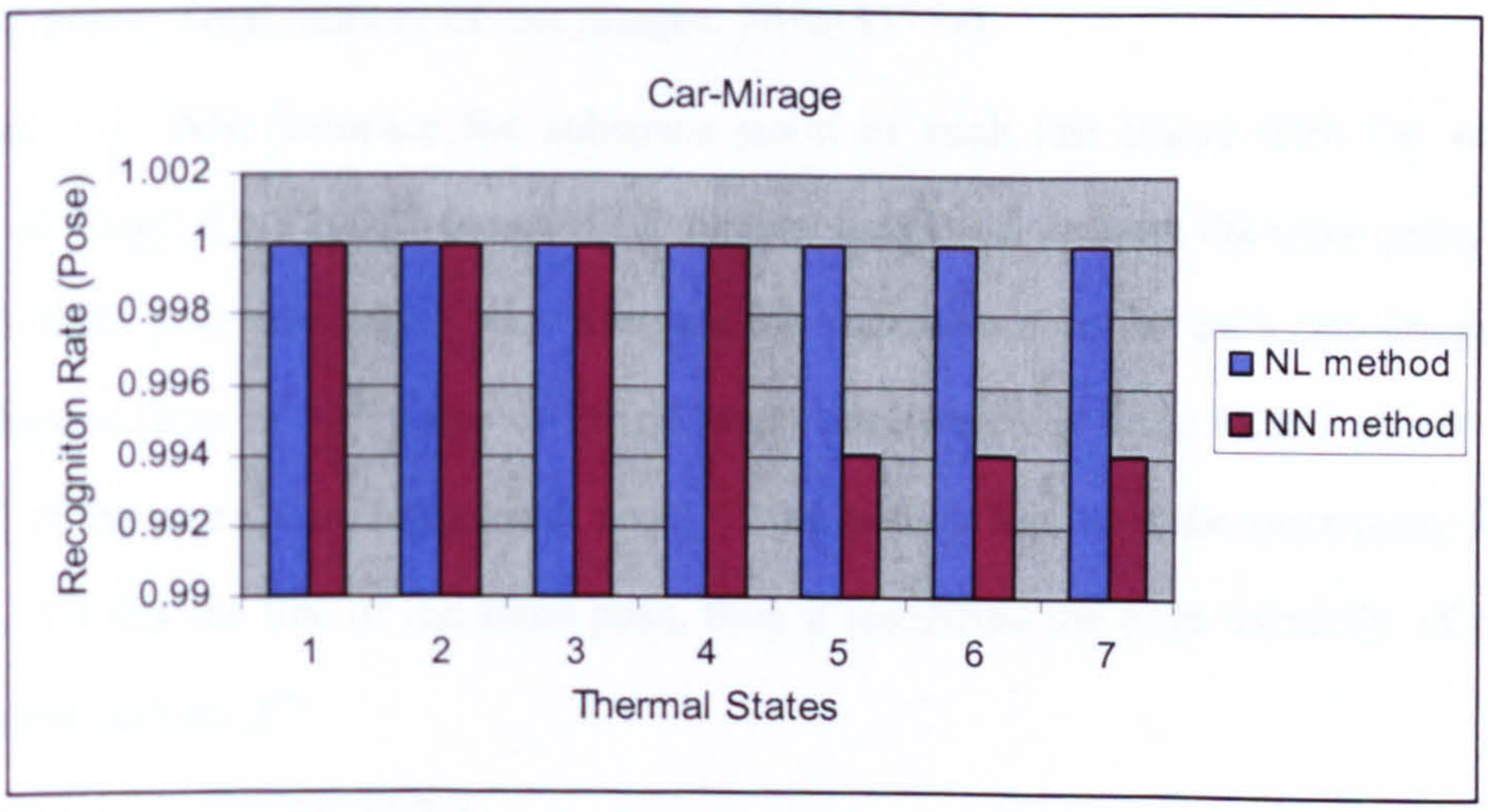


Figure 5-32 Pose Estimation Result of Car_Mirage

5.4.3 Testing Combined Effects of Thermal State Changing

Objective:

In the Landrover dataset, Tsig3 is a dataset representing thermal state changes in the Engine, Tsig4 represents the Grill, Tsig2 represents the Engine and Grill, which is a joint effect of Tsig3 and Tsig4. As stated in section 5.3.2, the vector formed by thermal states in Tsig2 is the sum of 2 vectors formed by thermal states in Tsig3 and Tsig4. This suggests us to predict some unknown thermal state which is a combination of two other individual thermal states.

In this experiment, we try to use Tsig3 and Tsig4 to predict the position in Eigenspace formed by Tsig2, and then compare the predicted Tsig2 with the real Tsig2 data.

Procedure:

Tests using the ground truth data:

Training set: Landrover, TSig2, TSig3 and TSig4 with three thermal states in each thermal signature (T0, T1 and T2), 337 poses[13]. Total number of training images: 3033 (337*3*3), note that not the whole training set is used in recognition stage, only TSig2 is used.

Test images: Landrover TSig2 with 8 thermal states(T3-T10) which is not used in training set, 337 poses; Total number of test images: 2696(337*8)

Method: **NN:** compare the subspace point of each test image with the subspace points of TSig2, T0-T2, 337 poses, if the nearest neighbour is from the same pose, then it identified the pose correctly. **NL:** compare the subspace point of each test images with the subspace lines of 337 poses of TSig2 (each line, which is fitted with 3 points T0, T1 and T2 of the same pose, represent a pose), if the nearest line is of the same pose, in other words, if it fits the line of the same pose, then it identified the pose correctly. (for detail of NL, see section..)[14]:

---

[13] Eigenspace of training set save as *ES_LandR_TSig2-4_T0-2.mat*

[14] This result saved in *LandR_Tsig2-4_T0-2_Tsig2_T3-10_Dire_20D.mat*, *LandR_Tsig2-4_T0-2_Tsig2_T3-10_NN_20D.mat*, and *LandR_Tsig2-4_T0-2_Tsig2_T3-10_Dire_predi_20D.mat*

Test using the predicted data:

Training set[15]: Landrover, TSig3-4, T0-2, 337 poses; Total number of training images: 2022 (337*2*3)

Test images: Landrover TSig2, T3-10, 337 poses; Total number of test images: 2696(337*8)

Method: **NL_Predi** – We use the subspace point cloud of TSig3 and TSig4 to predict the subspace points of TSig2, then use the NL method to do pose estimation using the predicted training set, that is, the vector formed by TSig2 is the sum of the vectors of TSig3 and TSig4.

Result and Discussion:

The results[16] of the comparison are shown in Figure 5-33. We see using the real date, the result from the NL method is better than the NN method. Using the predicted data, the NL_predi is still got good result though not as good as using the real data. Yet NL using the predicted data is better than the NN method using the real data as the object thermal state is very different from the thermal states in the training images.

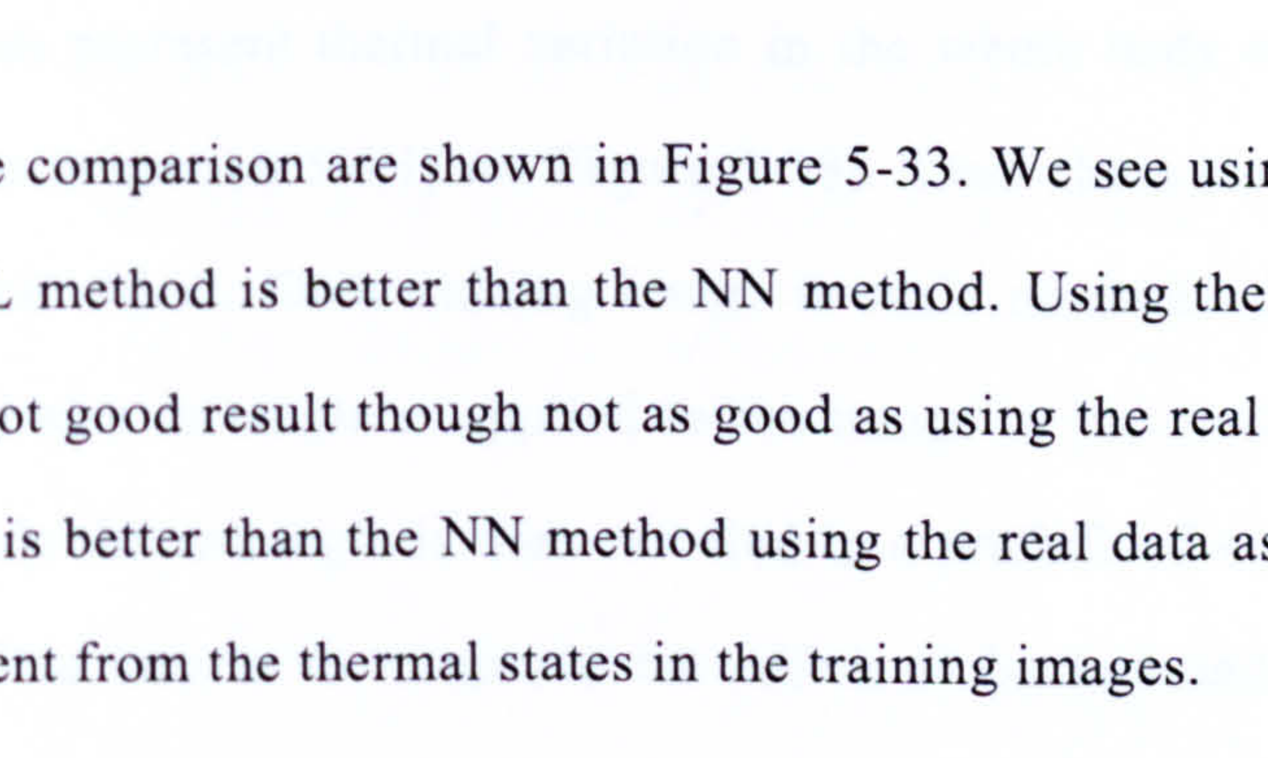


Figure 5-33 Recognition Result of NL_Predi method compare with NN and NL methods

[15] Eigenspace of training set save as *ES_LandR_TSig3-4_T0-2*
[16] This result is saved in *LandR_Tsig3-4_T0-2_Tsig2_T3-10_Dire_Perdi_20D.mat*

5.4.4 Tests on a large infrared scenes

Objective:

In this section, we test the proposed method on large infrared scenes including suburban areas and city areas to see it's performance on real application. The objects of interest in these scenes are in different scales and with clutter and occlusion effects. Eight scenes with different number of vehicles are tested.

Procedure:

In the training stage, we train the 11 objects (see Figure 3-38) separately to form 11 object Eigenspaces. For each object, we use 337 poses and 3 thermal states for each pose. The 3 thermal states represent thermal variation in the whole body of the vehicle (same thermal variation as in section 5.4.1, see Figure 5-18). Thus, there are 1101 images in the training set for each object. Each training image is scale normalized (see Figure 5-34), that is, a minimum size rectangle is applied to the image to just cover the whole object area. The area inside the rectangle is then rescaled to a predefined scale. In this test, the scale is 64×64. This step is to minimize the effect of background in the recognition performance. No energy normalization is used.

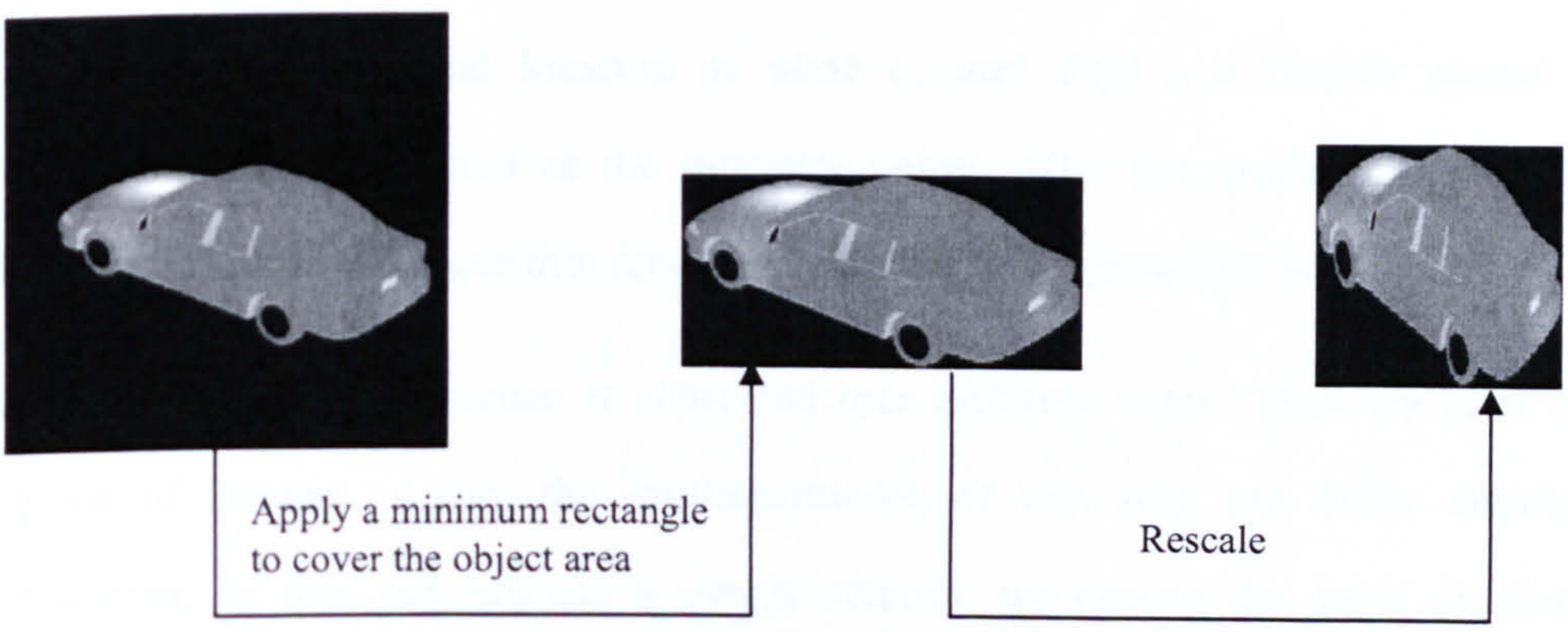


Figure 5-34 Illustration of Scale normalization

In the recognition stage, given an image of a scene, first, a segmentation process is used to define different areas in the scene. The segmentation method can be chosen depend on different applications. In this test, we selected a segmentation method with *Normalized*

Cuts [141] [143]. The approach treats image segmentation as a graph partitioning problem. That is, the set of points in an arbitrary feature space are represented as a weighted graph $G=(V,E)$, where the nodes of the graph are the points in the feature space, and an edge is formed between every pair of nodes. The weight on each edge $w_{i,j}$ is a function of the similarity between the two nodes i and j. In grouping, it seeks to partition the set of vertices into disjoint sets, where by some measure the similarity among the vertices in a set is high and across different sets is low. The traditional *cut* based method [144] is to partition a graph into k-subgraphs such that the maximum cut across the subgroups is minimized, where the *cut* is: $cut(A,B)=\sum_{u\in A,v\in B} w_{u,v}$. The difference in *Normalized Cuts* is that it computes the cut cost as a fraction of the total edge connections to all the nodes in the graph to avoid the local optimization: $Ncut(A,B)=cut(A,B)/assoc(A,V)+cut(A,B)/assoc(B,V)$. In the implementation, the graph $G=(V,E)$ is constructed by taking each pixel as a node and the edge weight $w_{i,j}$ between nodes i and j are defined as the product of a feature similarity term and spatial proximity term:

$$w_{i,j}=e^{\frac{-\|F(i)-F(j)\|_2^2}{\sigma_I}}*\begin{cases} e^{\frac{-\|X(i)-X(j)\|_2^2}{\sigma_X}} & if \quad \|X(i)-X(j)\|_2<r \\ 0 & otherwise \end{cases} \tag{5-15}$$

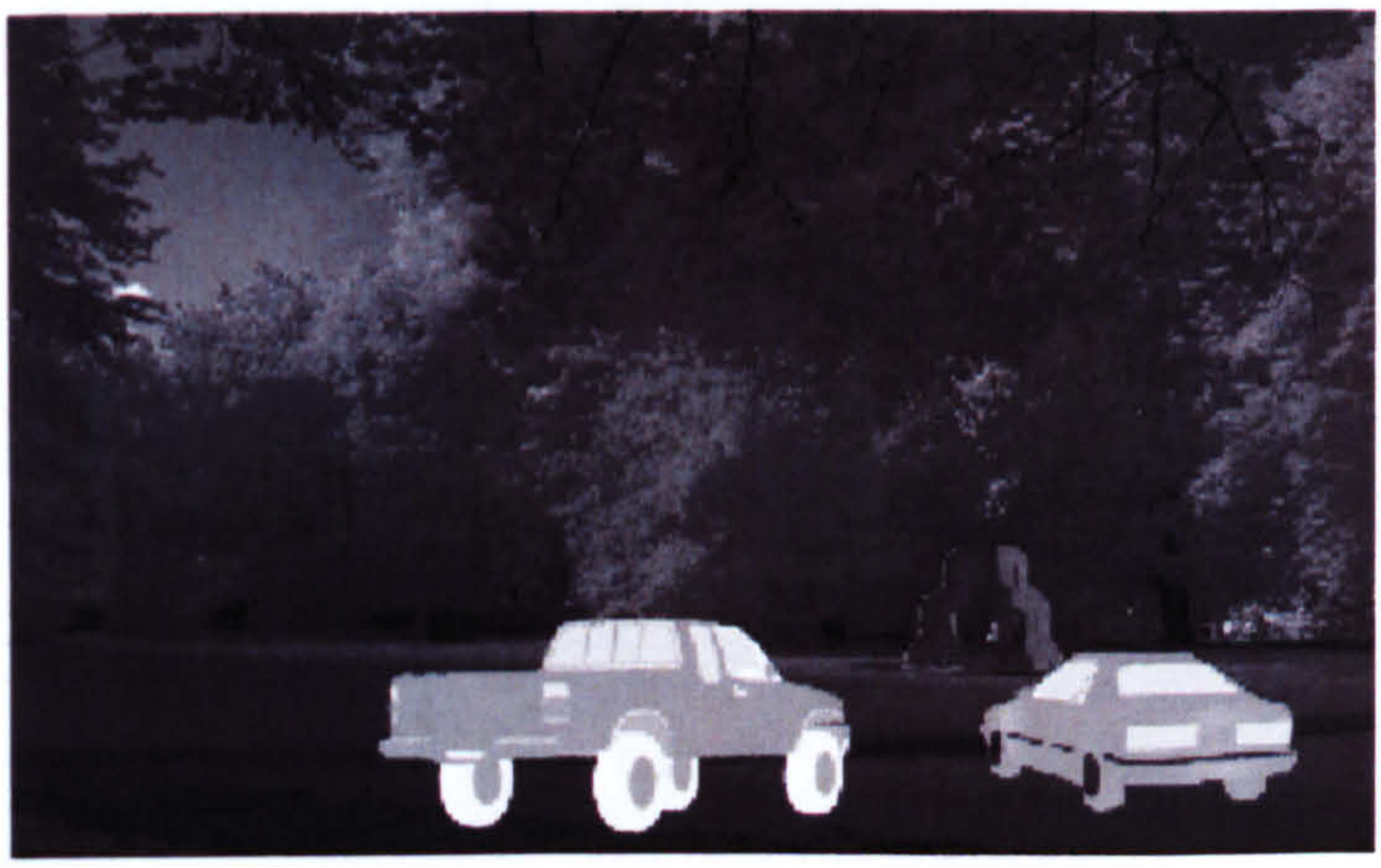
where $X(i)$ is the spatial location of node i, and $F(i)$ is a feature vector. In this experiment, $F(i)$ is defined as the intensity value. After the graph is broken into two pieces, we can run the algorithm recursively on the two partitioned parts.

After segmentation, the scene is separated into different areas. Then we have to select regions of interest. Again, the implementation of this step can differ dependent on application. In this test, we use a simple criteria: we choose the areas smaller than a predefined scale which is decided by the size of the objects of interest and the camera position. Given the area of interest, we define the smallest rectangle which contains it and then rescale the rectangular image to the predefined scale in the training set, 64×64. Finally, we compare this image to our 11 object Eigenspaces. The two outputs from the

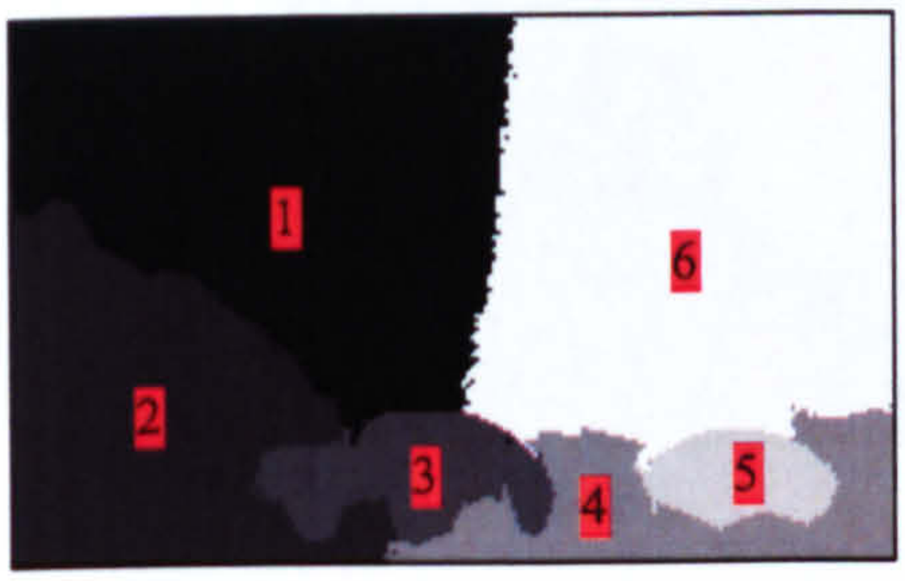
object Eigenspace recognition decide the object identification: *Distance to the nearest line* (in-space error) and *Distance from the Eigenspace* (out-of-space error). Two thresholds can be defined for these two outputs. The object Eigenspace which has the outputs below the thresholds represent the identified object. However, the thresholds depend on many factors, e.g., quality of the image, scale of the object, training image sets, etc.. In this test, instead of defining thresholds, we firstly find the smallest *Distance from the Eigenspace* to identify object. Then, if the corresponding *Distance to the nearest line* is in the three smallest ones, the decision is accepted. If not, the area is identified as containing no object of interest. At the same time, we use the NL method to recognize poses. In the scene where cluttering and occlusion appears, robust sampling is applied.

Result and Discussion:

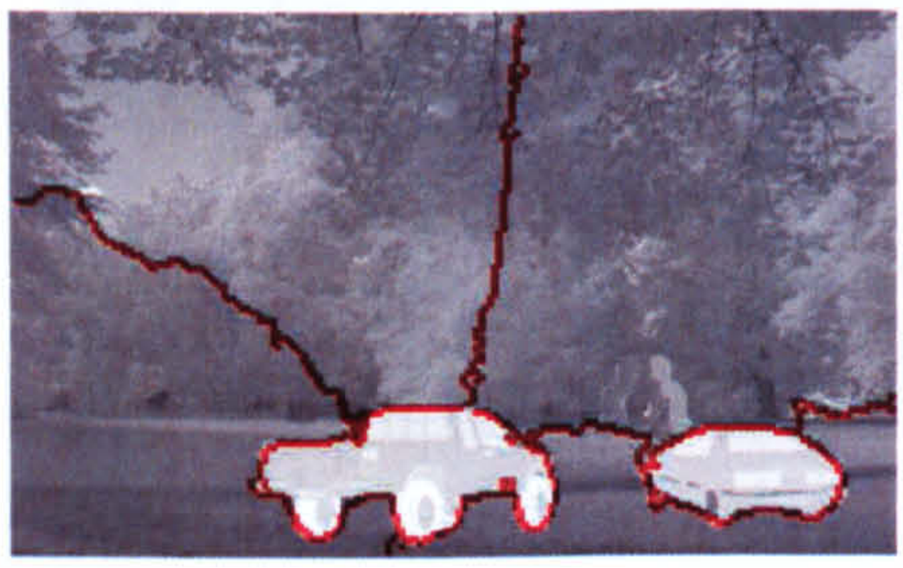
Figures 5-31 to 5-37 show the result of object recognition and pose estimation. Figures 5-31, 5-32, and 5-36 are scenes in which one or two vehicles are in the cluttered environment with roads, trees and houses. We see that in these scenes, although sometimes the segmentation process makes small errors (e.g., Figure 5-36 (d) and Figure 5-40(d)), the vehicles are recognized at the correct object and pose. In Figure 5-39 and Figure 5-37, there is one vehicle in the scene occluded by a tree or by another vehicle. We see that in this case, one of the objects was recognized as the right object but wrong pose. Figures 5-34 and 5-37 show more complex scenes. In Figure 5-38, three cars are in a residential area, in which a window is selected as one of the interesting areas. We see that the algorithm can deduce that this does not contain any object in the database. However, the algorithm sometimes identifies some area containing an object as false alarms (Figure 5-41 (f)).



(a)



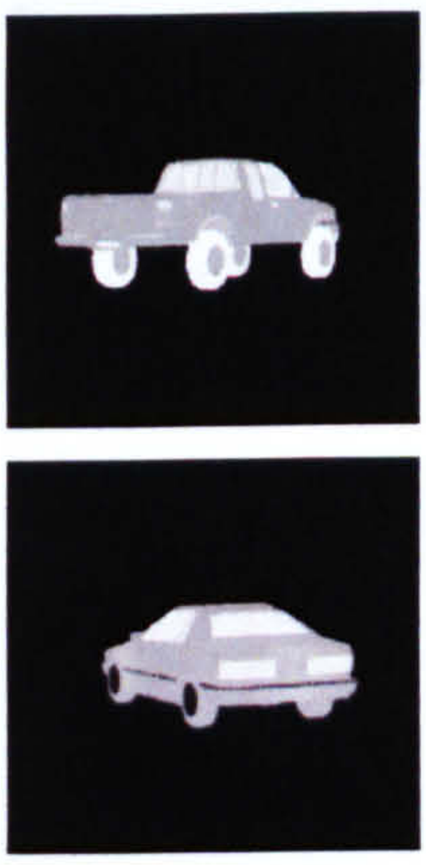
(b)



(c)

(d)

Recognize as
obj 11 pose 68



(f)

(e)

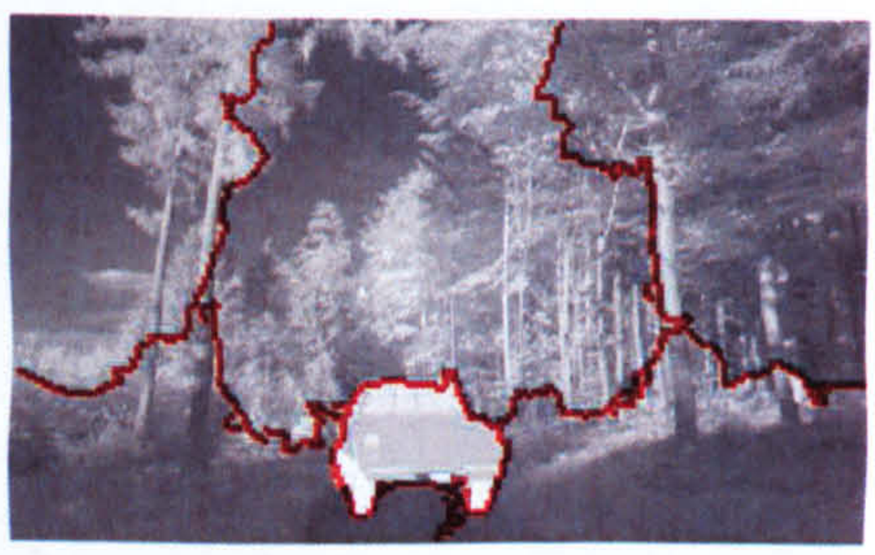
Recognize as
obj 9 pose 265

(g)

Figure 5-35 Scene_1, (a) Original Scene, (b) (c) Segmentation of the scene, (d) (e) interested areas – 3 & 5, (f) (g) recognition results

(a)

(b)



(c)

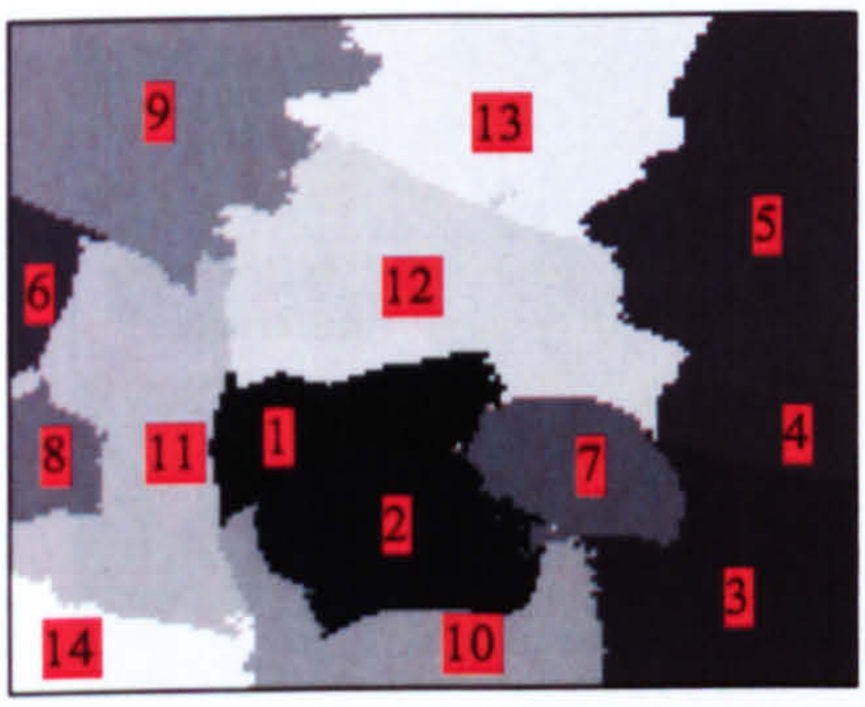
(d)

Recognize as
obj 11 pose 210

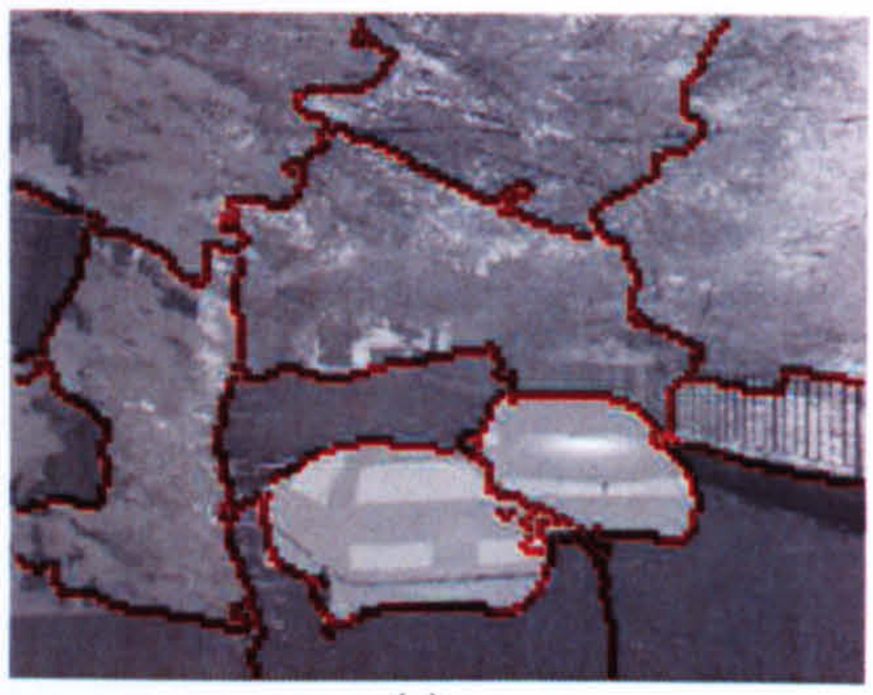
(e)

Figure 5-36 Scene_2, (a) Original Scene, (b) (c) Segmentation of the scene, (d) interested area - 3, (e) recognition results

(a)



(b)

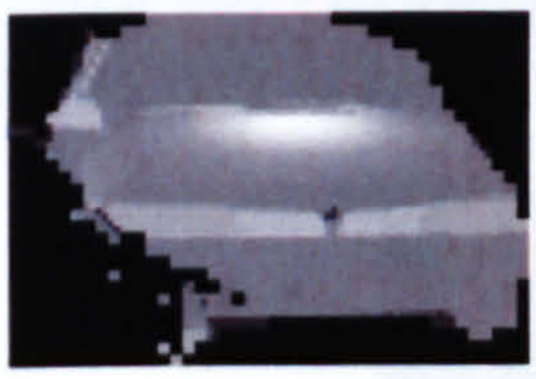


(c)

(d)

Recognize as
obj 9 pose 74

(f)



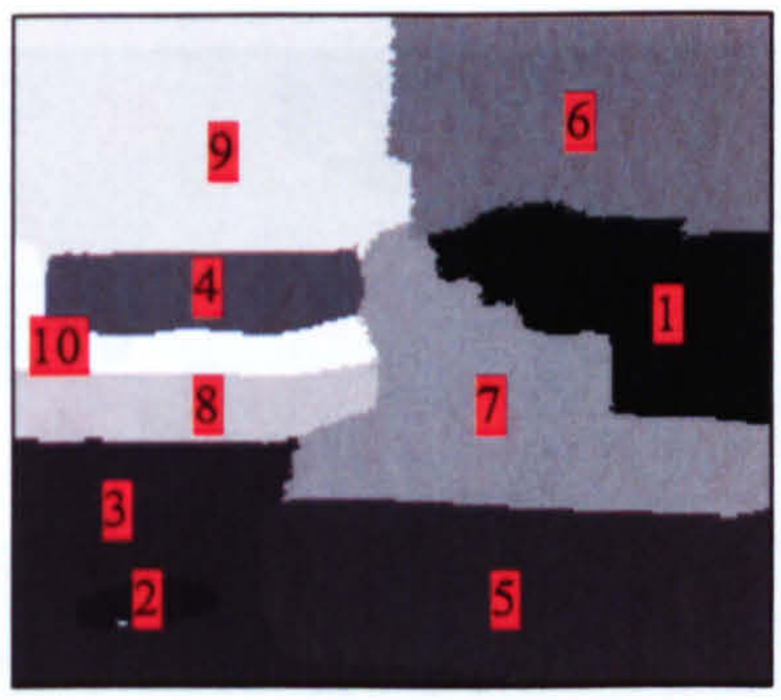
(e)

Recognize as
obj 5 pose 89

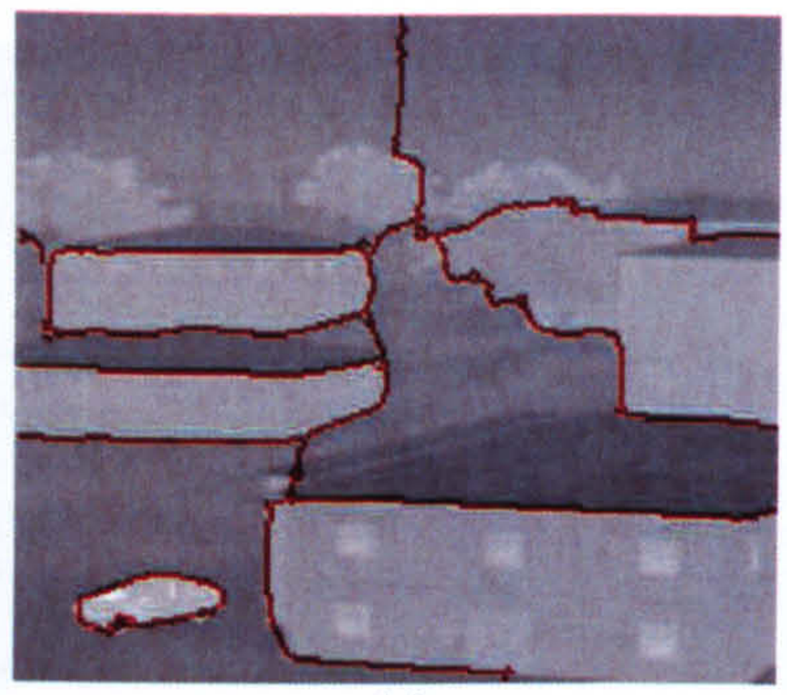
(g)

Figure 5-37 Scene_3, (a) Original Scene, (b) (c) Segmentation of the scene, (d) (e) interested areas – 2 & 7, (f) (g) recognition results

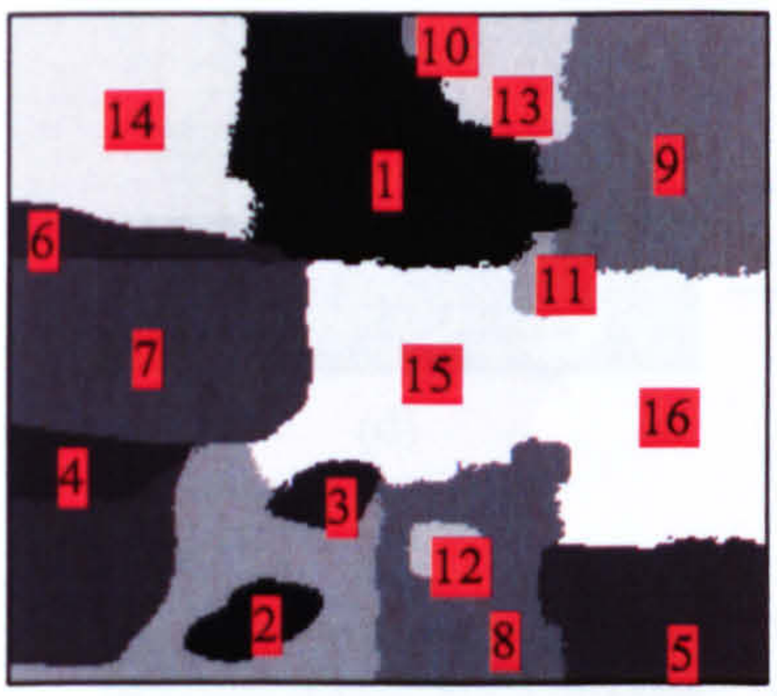
(a)



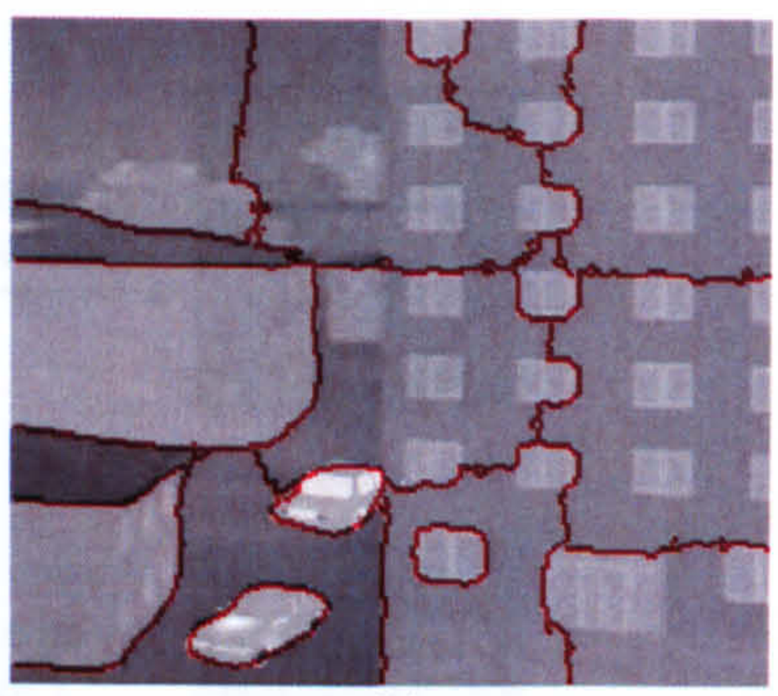
(b)



(c)



(d)

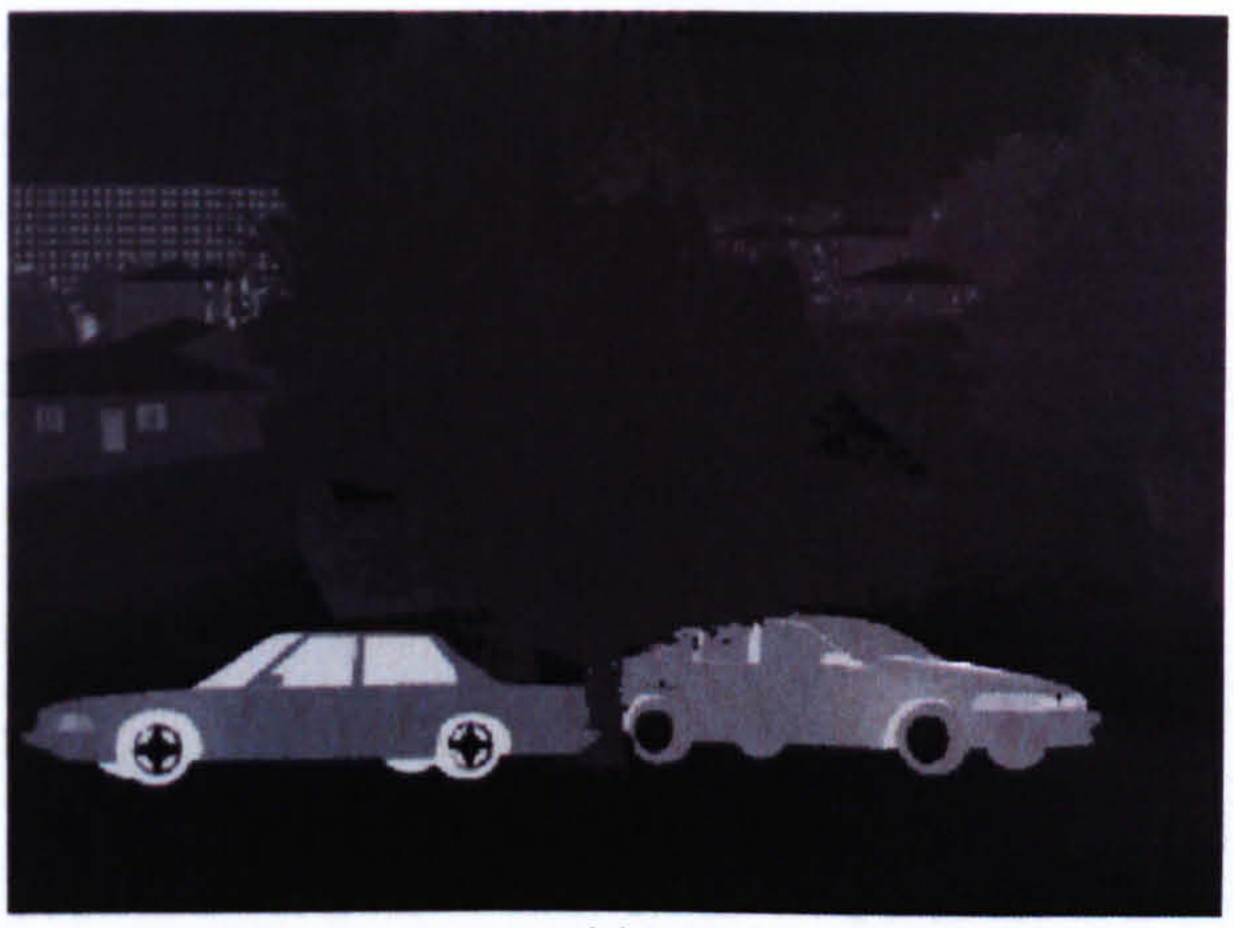


(e)

(f) (j) (h) (m)

(g) (k) (i)

Figure 5-38 Scene_4, (a) Original Scene, (b) (c) Segmentation of left half image, (d) (e) Segmentation of right half image, (f) (g) (h) (i) interested areas – 2(left), 2(right), 3 & 12, (j) (k) (m) recognition results



(a)

(b)

(c)

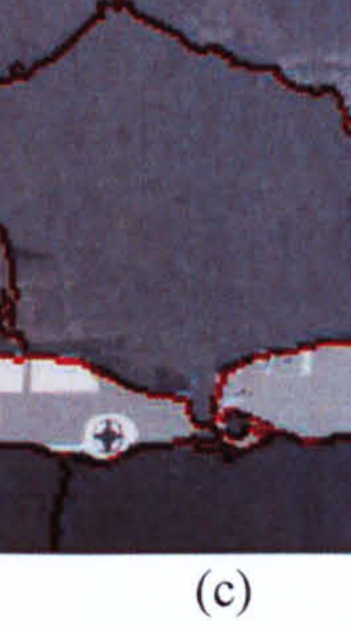
(d)

Recognize as
obj 8 pose 220

(f)

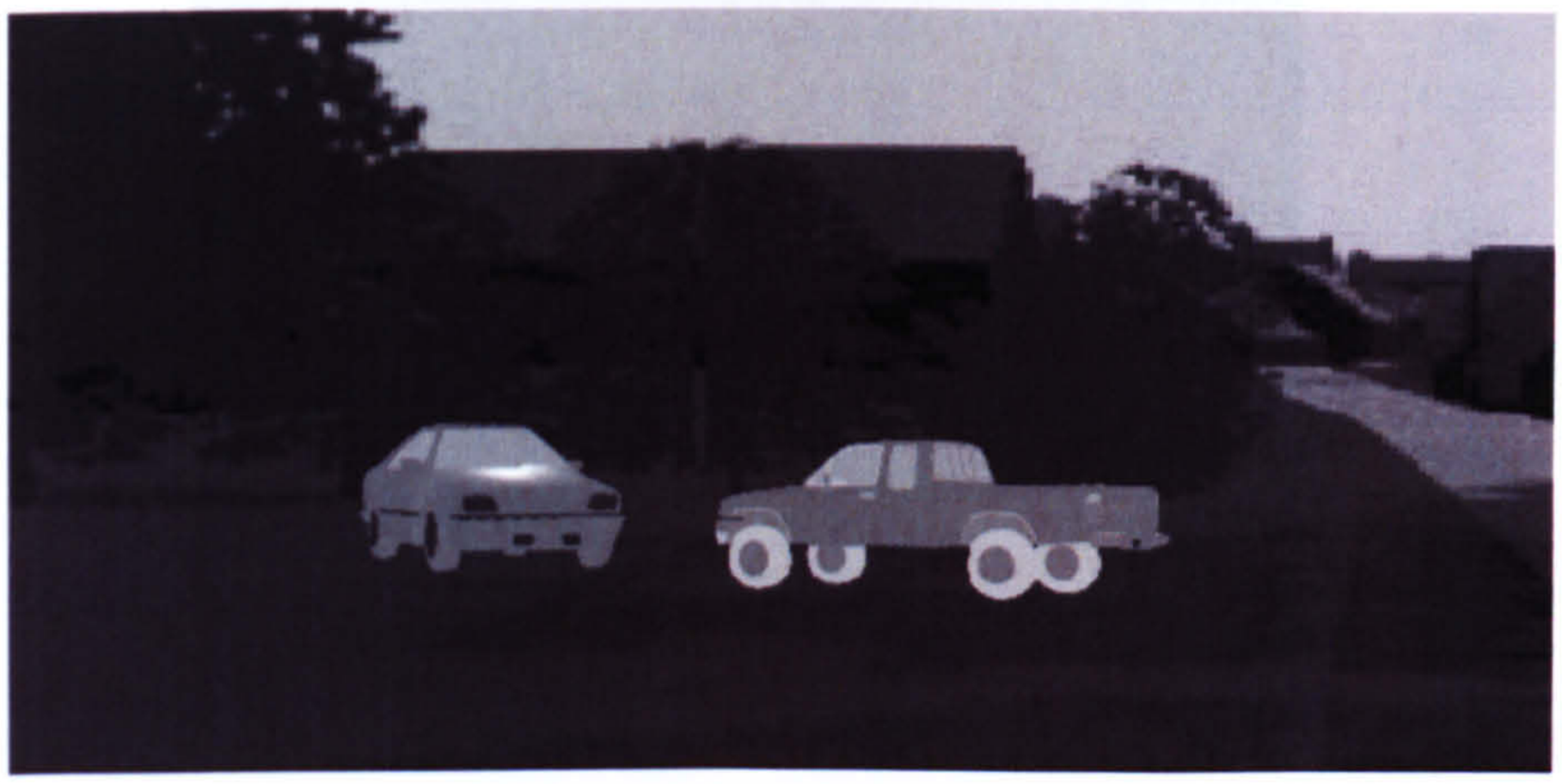
(e)

Recognize as
obj 5 pose 1

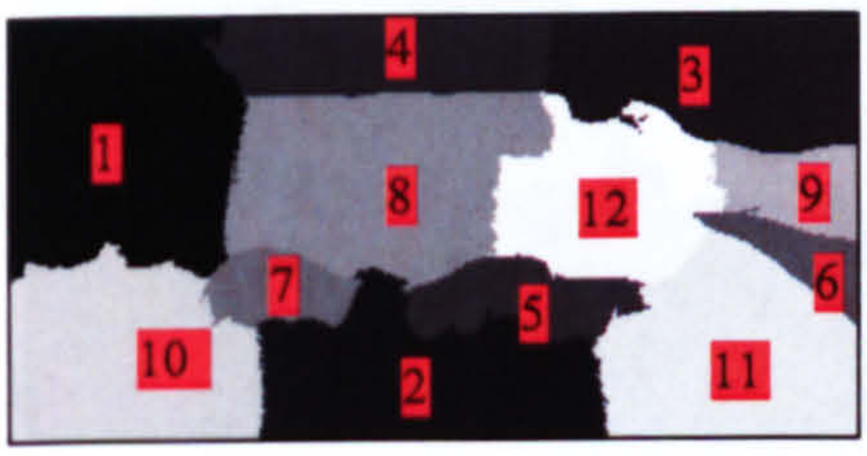


(g)

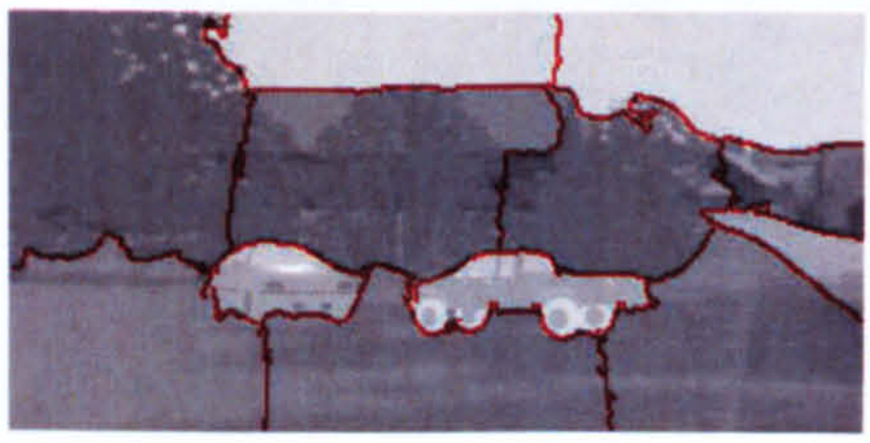
Figure 5-39 Scene_5, (a) Original Scene, (b) (c) Segmentation of the scene, (d) (e) interested areas – 2 & 4, (f) (g) recognition results



(a)



(b)



(c)

(d)

Recognize as
obj 9 pose 220

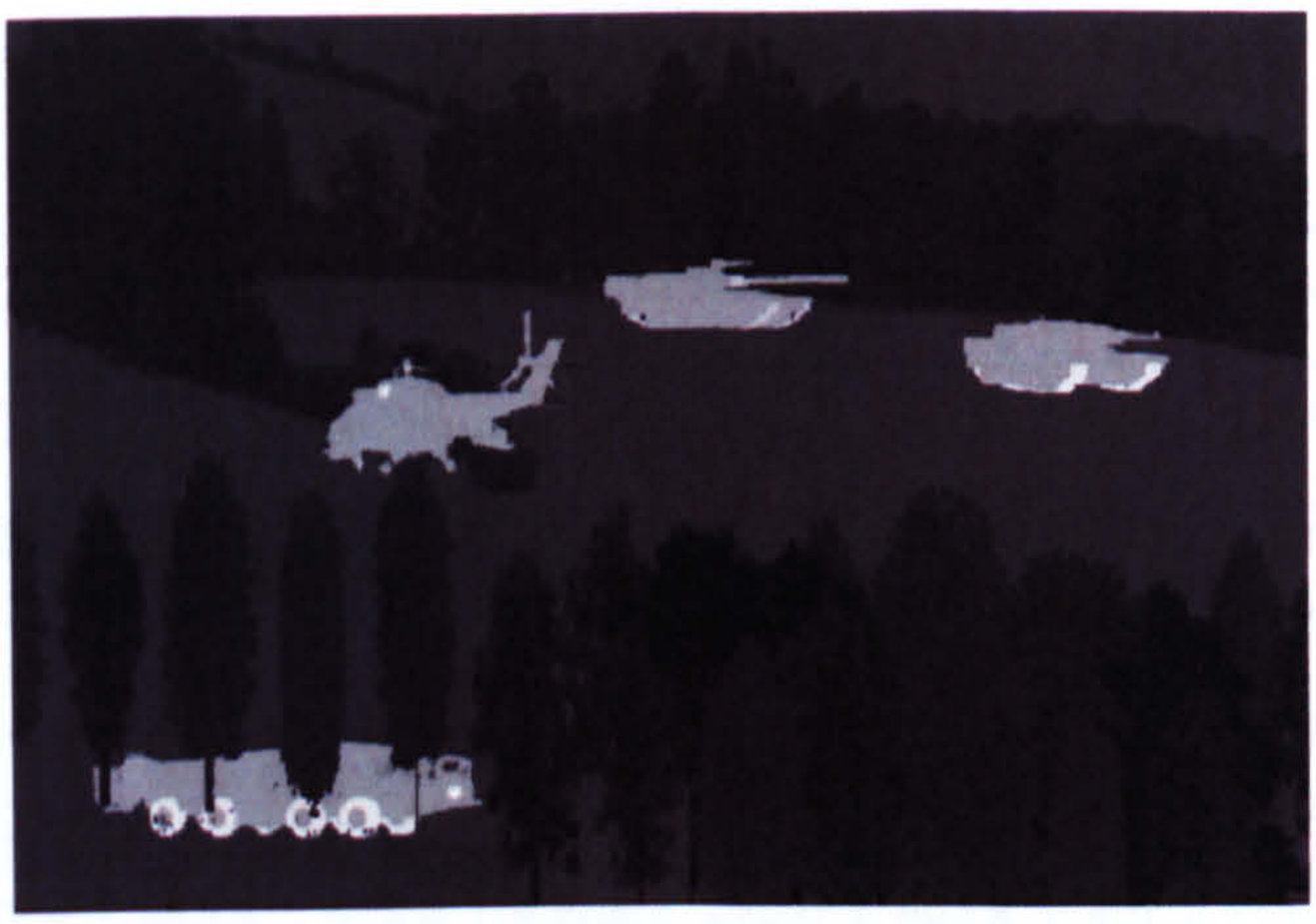
(f)

(e)

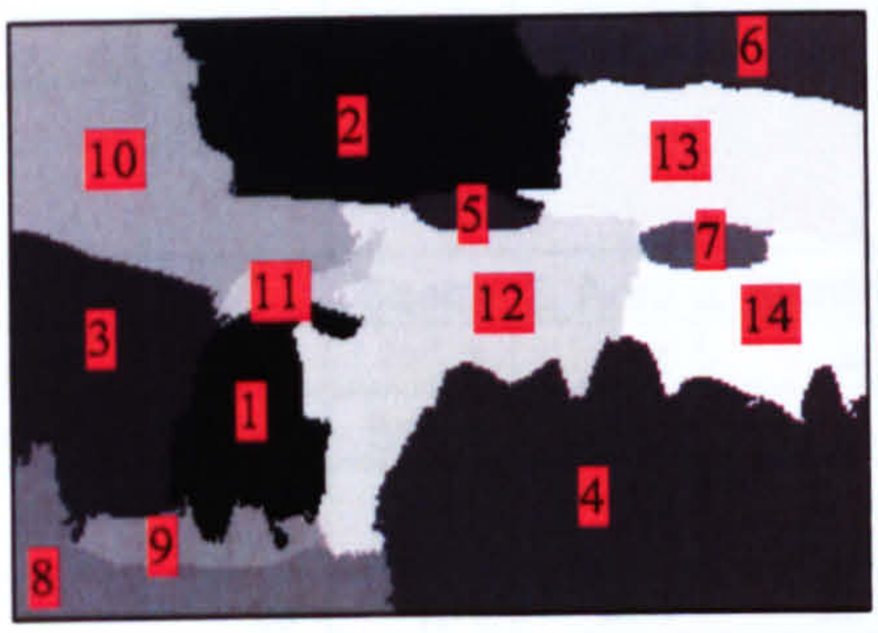
Recognize as
obj 11 pose 1

(g)

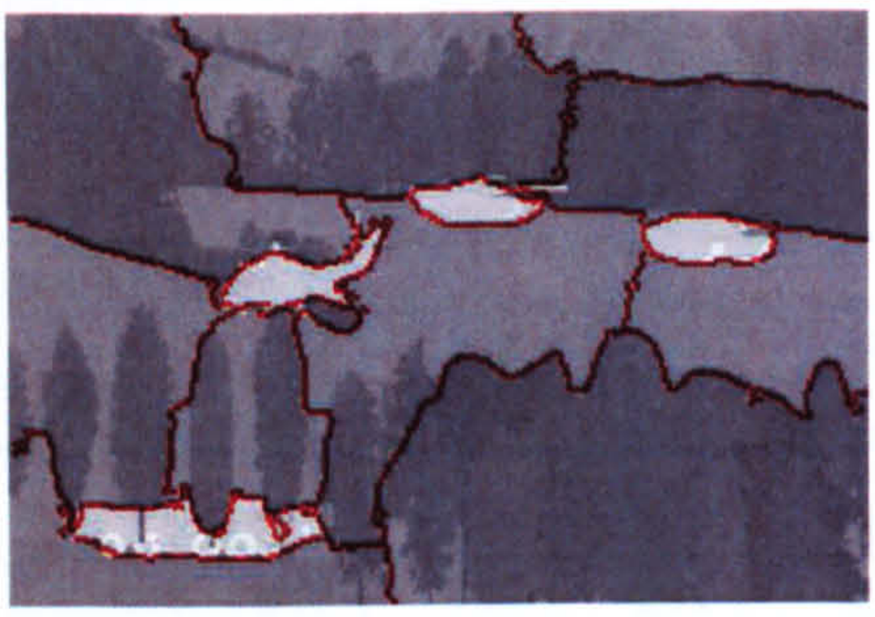
Figure 5-40 Scene_6, (a) Original Scene, (b) (c) Segmentation of the scene, (d) (e) interested areas – 5 & 7, (f) (g) recognition results



(a)



(b)

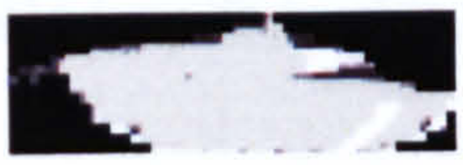


(c)

(d)

Recognize as
obj 1 pose 63

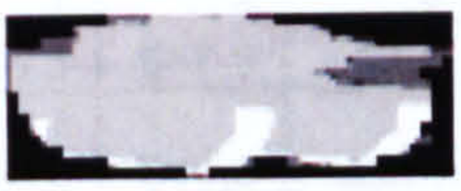
(h)



(e)

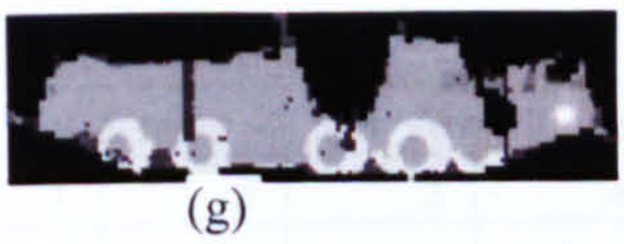
Recognize as
obj 4 pose187

(i)



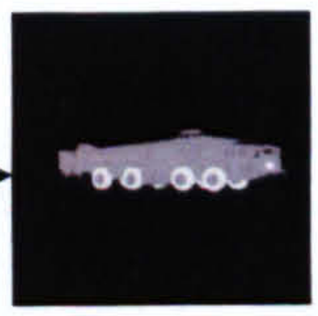
(f)

Not recognized as
any object



(g)

Recognize as
obj 7 pose 1



(j)

Figure 5-41 Scene_7, (a) Original Scene, (b) (c) Segmentation of the scene, (d) (e) (f) (g) interested areas – 11, 5, 7, & 9 (h) (i) (j) recognition results

Scene_1, Area 3, Identified as object 11, pose 68											
Object Eigenspace	1	2	3	4	5	6	7	8	9	10	11
Distance to the line	573.6	456.5	475.2	501.7	461.9	587.7	514.7	477.7	460.8	482.2	194.4
Distance to Eigenspace	709.5	471.3	500.3	541.3	446.9	468.8	462.5	423.6	441.2	492.6	278
Pose	34	119	82	32	255	7	234	146	251	111	68

Scene_1, Area 5, Identified as object 9, pose 265											
Object Eigenspace	1	2	3	4	5	6	7	8	9	10	11
Distance to the line	572.4	445.4	499.3	466.9	240.8	463.2	467.3	403.4	229.7	392.9	451.8
Distance to Eigenspace	578.6	420.3	441.2	381.6	287.9	353.1	351.2	307.6	282.1	347.8	322.9
Pose	67	310	117	32	265	7	132	189	265	114	257

Scene_2, Area 3, Identified as object 11, pose 210											
Object Eigenspace	1	2	3	4	5	6	7	8	9	10	11
Distance to the line	490.9	516.4	502.2	512.3	428.9	512.9	429.3	417.4	367	494.2	311.8
Distance to Eigenspace	601.7	463.1	434.1	426.3	387.8	386.7	422.5	385.4	392.7	399	366.6
Pose	99	101	256	32	254	7	210	155	254	20	210

Scene_3, Area 2, Identified as object 9, pose74											
Object Eigenspace	1	2	3	4	5	6	7	8	9	10	11
Distance to the line	554.1	522.2	510.9	450.2	383.4	471.5	460.2	426.4	295.7	423.3	416.1
Distance to Eigenspace	599.9	511.6	445.9	353	310.7	339.4	361.6	310.5	306.7	328.8	332.5
Pose	217	29	188	142	74	142	252	191	74	119	256

Scene_3, Area 7, Identified as object 5, pose89											
Object Eigenspace	1	2	3	4	5	6	7	8	9	10	11
Distance to the line	556.3	587.3	513.6	475.8	451	493.3	505.6	475.8	425.3	446.4	464
Distance to Eigenspace	606.2	540.3	522.4	389.5	335.5	382.8	385.1	341.3	345.4	391.9	345.7
Pose	231	108	260	119	89	109	132	173	74	119	256

Scene_4, left, Area 2, Identified as object 5, pose192											
Object Eigenspace	1	2	3	4	5	6	7	8	9	10	11
Distance to the line	598.5	505.3	603.3	650.3	485.9	580.7	633.7	536.5	488.8	484.4	627.2
Distance to Eigenspace	523.9	510.1	589.7	421.9	302.7	388.8	377.5	338.3	307.6	412.9	346.9
Pose	211	43	78	121	192	115	58	268	192	111	66

Scene_4, right, Area 2, Identified as object 5, pose22											
Object Eigenspace	1	2	3	4	5	6	7	8	9	10	11
Distance to the line	747.6	687.3	648.3	645.6	596.4	568.9	641.6	626.5	618.1	418	678.3
Distance to Eigenspace	662.9	556.5	589.4	367.6	308.3	368.7	356	315.6	324	355.2	392.2
Pose	173	26	173	111	22	316	237	274	22	314	224

Scene_4, right, Area 3, Identified as object 8, pose43											
Object Eigenspace	1	2	3	4	5	6	7	8	9	10	11
Distance to the line	773.5	575.4	583.1	753	567.7	600	557.8	538.2	537.3	371.1	631.1
Distance to Eigenspace	562.7	496.9	584.2	461.6	446.2	437.5	423.5	413.2	440.6	463.9	419.7
Pose	122	134	215	113	69	33	237	43	69	225	236

Scene_4, right, Area 12, Identified as not from the object database											
Object Eigenspace	1	2	3	4	5	6	7	8	9	10	11
Distance to the line	673.6	625.2	585.6	647.8	403.3	629.1	347.1	379.7	384.7	288.8	438.6
Distance to Eigenspace	638.8	676.7	603.6	368	330.4	354.8	352.1	346.5	326.7	359	350.8
Pose	48	119	235	32	255	7	71	39	32	7	9

Scene_5, Area 2, Identified as object 5, pose 1											
Object Eigenspace	1	2	3	4	5	6	7	8	9	10	11
Distance to the line	515	453.9	398.5	414.1	317	441.2	438.9	399.5	367.4	467.7	480
Distance to Eigenspace	603.6	479.7	493.6	425.2	327.5	381.9	382.5	352.5	333.9	388.1	374.3
Pose	117	251	59	66	1	34	252	69	148	119	61

Scene_5, Area 4, Identified as object 8, pose 220											
Object Eigenspace	1	2	3	4	5	6	7	8	9	10	11
Distance to the line	508.5	422.9	443.2	456.3	399.7	500.9	440.2	229.9	357.2	477.8	489.8
Distance to Eigenspace	599.7	402.9	421.2	451.9	374.8	368.4	384.9	295.6	350.4	379.4	399.3
Pose	83	67	108	254	51	222	228	220	54	125	66

Scene_6, Area 5, Identified as object 11, pose 1											
Object Eigenspace	1	2	3	4	5	6	7	8	9	10	11
Distance to the line	582.1	477.4	553	471.5	437.5	502.5	471.2	479.2	413.5	471.1	227.3
Distance to Eigenspace	628.2	503.1	438.3	472	443.5	421.2	435.4	413.6	427.3	436.4	301.4
Pose	110	36	217	70	70	241	242	181	70	137	1

Scene_6, Area 7, Identified as object 9, pose 220											
Object Eigenspace	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**
Distance to the line	532.5	469.3	428.9	462.4	410.9	518.4	406.5	354.3	353	336.5	457.9
Distance to Eigenspace	585.4	425.1	500.7	384.6	305.8	334	337.7	294.4	290.8	339.8	343
Pose	73	119	273	216	243	142	255	45	220	212	61

Scene_7, Area 5, Identified as object 4, pose 187											
Object Eigenspace	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**
Distance to the line	803.9	687.5	690.6	548.2	629.9	603.6	629.8	626.5	630.5	547.1	639.7
Distance to Eigenspace	728.5	641.5	695.8	406.8	469.5	430.5	486.6	467.4	458	414	465.7
Pose	21	278	233	187	212	258	253	49	266	135	38

Scene_7, Area 7, Identified as not from the object database											
Object Eigenspace	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**
Distance to the line	880.5	764.8	691	867.4	483	753.7	579.7	520.6	480.3	541.7	621.3
Distance to Eigenspace	763.4	605.5	586.6	495.1	450.9	449	466.9	450.8	435.2	427.7	458.5
Pose	211	168	119	248	255	171	5	156	255	213	211

Scene_7, Area 9, Identified as object 7, pose 1											
Object Eigenspace	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**
Distance to the line	687.5	579.6	598.4	527	622.8	564.4	365.7	616.7	619.7	472.2	686.6
Distance to Eigenspace	768.9	698.4	632.2	648.1	679.3	628	617	688.1	690.3	634.1	660
Pose	271	133	31	334	57	231	1	116	102	240	8

Scene_7, Area 11, Identified as object 1, pose 63											
Object Eigenspace	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**
Distance to the line	301.9	733.3	721.3	527	622.8	564.4	365.7	616.7	619.7	472.2	686.6
Distance to Eigenspace	439.2	768.9	709.5	648.1	679.3	628	617	688.1	690.3	634.1	660
Pose	63	271	139	334	57	231	1	116	102	240	8

5.4.5 Conclusion

In the experiments in this chapter, we compared the proposed NL method to the NN method on infrared images. In each experiment, the objects in test images are in different thermal states from the training set. Results show that in all of these tests, the proposed NL method gets a better recognition result compared to the original method. For pose

estimation in infrared images, the NL method achieves a higher recognition rate than the original method for all objects with and without noise.

Using the proposed method, we could also predict the joint effect (multi-part variation) of two known thermal signatures of single-part thermal state variation. The resultant thermal signature is also a vector in appearance space which is calculated by the sum of the two vectors representing single-part variation. In the experiment, we also tested the accuracy of this prediction. Results show that the recognition result using those predicted thermal states with the NL method is even better than using the real data with NN(original) method.

In tests on large infrared scenes which contain one or more object, we employ the normalized cut segmentation algorithm to segment the scene and then select the areas of interest. We find that generally the proposed method could in cope with different thermal state, different scale of the object, clutter and small scale occlusions, although sometimes the object was recognized as another similar pose of another object.

Chapter 6

Conclusion and Future Work

6.1 Summary of Contributions

The main difference between object recognition in infrared and in visible imagery is that in infrared imagery, the object's thermal states will affect the recognition result. Thus, modelling the infrared signature of the objects is one of the key issues in designing a recognition system.

- **One of the main contributions** of this thesis is that we propose a method to model changes in thermal states of objects using appearance. We model possible thermal state changes of an object by combinations of several single component changes. For several images among which there are single component changes, their projection in Eigenspace can be approximated as a single direction. The combination of two single component changes is proved to be the diagonal of the parallelogram formed by the two direction lines of the two single component changes. Hence, we proposed an algorithm to predict the subspace projection of any new image. With this prediction as part of the model, we could recognize objects in new thermal states. A nearest line classifier is used in the recognition stage. For pose estimation in infrared images, the new method achieves a higher recognition rate for all new thermal states than the original method.

- **The other main contribution** is to provide a probabilistic framework as an extension of the general appearance based method. With this framework, the form of the

recognition result is not only the object identification, but also a confidence in that judgment. The probability is determined by two facts: in-space-error and out-of-space-error. This first is calculated by a Gaussian model of the distance between a input point and the nearest sampled point on manifold. The second is measured by the recovery error of image pixels. Using this probabilistic framework, we set an 'image window' on the test image and adjust the position of the window. The 'image window' method allows the system to bear small in-plane transformations of the object in the test image and recognize poorly segmented test images. With this method, the recognition rate is improved by an average of more than 15% for poorly segmented images.

The thesis has also addressed the following other issues:

- To strengthen the appearance-based model we interpolate the discrete points in Eigenspace to form a manifold. The cubic spline is chosen as the interpolation method. This increases the recognition rate for noisy images.

- To make the algorithm more efficient, a k-d tree search algorithm was used in recognition stage to search for the nearest point. Results show more than half of the time is saved using this search algorithm when compared with exhaustive search.

- To recognize images having noise and occlusion, the robust sampling algorithm is used to effectively select a subset of non-corrupted pixels from the whole test image. Compared to the standard method, the random sampling method improves the result considerably, e.g., the projection distance is low and stable when up to 50% area of the image is randomly corrupted. For infrared images with clutter, e.g., trees in the background, trees in front of the object, or the object occluded by other objects, the method could successfully identify the object in these scenes.

- We also questioned the data decomposition method, PCA, as the basis of the appearance based recognition. We analyze and test a non-linear dimensionality reduction method, Isomap. We find that Isomap works well on relatively simple datasets where the distance between samples in their original space reflects the distance in their observation

space. In more complex datasets, compared to PCA, Isomap only works better when using fewer dimensions. Only in applications in which reduced number of dimensionalities is the most important consideration is Isomap recommended.

6.2 Future Work

A major assumption in the appearance based method is that a good segmentation exists. In this thesis, we proposed a method to solve one problem caused by bad segmentation, small in-plane transformations. However, bad segmentation could cause other problems. For example, in appearance-based methods, the scale normalization procedure is achieved by adjusting the object size such that one dimension of the object reaches the border of the image, to make sure that the object in the training image is the same size as in the test image. This procedure can be corrupted when a bad segmentation is made. One future task is to find or develop a proper segmentation method and combine it with the current recognition system.

In the current method, each image is firstly represented as a vector by scanning the image up and down, left to right. From the experiments of small in-plane transformations, we see that the manner of scanning affects the recognition result, e.g., vertical small in-plane transformation causes worse recognition result than horizontal one. A comparison of different manners of scanning will be interesting. For square image, the common ways of scanning are by row and by column. Other way of scanning, e.g., from centre to the border, etc., could be an alternative. The scan can be varying depends on the nature and shape of the objects of interest or be guided by some geometric features of the objects.

The proposed method to model the changes in thermal state consider the major source which contributes to the appearance of the object in infrared imagery, the thermal emission from the body and surface of the objects. However, thermal reflection also affects the appearance in an infrared image, especially in daytime. Thus, another expansion of the current system is to include a model of thermal reflection. The main

requirement to model thermal reflection is research on modelling illuminations within an appearance based method.

Appendix A

Fundamentals of infrared Imaging

One significant advantage of using infrared imagery is that infrared sensors are to a lesser extent dependent on different weather and illumination conditions than visible wave sensors: even day or night snapshots of the same scene are very similar, thus reducing the range of situations to be taken into account. This is because the majority of the captured intensity, at least at longer wavelength, is from direct thermal emission. In this chapter, we review the theory of this thermal emission/radiation and discuss the atmospheric window which explains why infrared sensing is of particular interest in a remote sensing scenario. A short discussion of infrared sensors is included. Finally, we review some pre-processing necessities for infrared imagery, e.g. radiometric calibration for data normalization and temperature mapping.

A.1 infrared Radiation

Infrared radiation lies between the visible and microwave portions of the electromagnetic spectrum (see Figure A - 1). There is no clear-out sub-dividing line within infrared radiation [145][146][147][148]. For the sake of our later discussion, we choose a division [2] as that most related to the different types of current infrared sensors: near infrared (0.75-3um), middle infrared (3-6um), far infrared (6-15um), and very far infrared (>15um).

The primary source of infrared radiation is heat or thermal radiation. This is the radiation produced by the motion of atoms and molecules in an object. The higher the temperature, the more the atoms and molecules move and the more infrared radiation they produce. Any object which has a temperature i.e. anything above absolute zero (-459.67 degrees Fahrenheit or -273.15 degrees Celsius or 0 degrees Kelvin), radiates in the infrared.

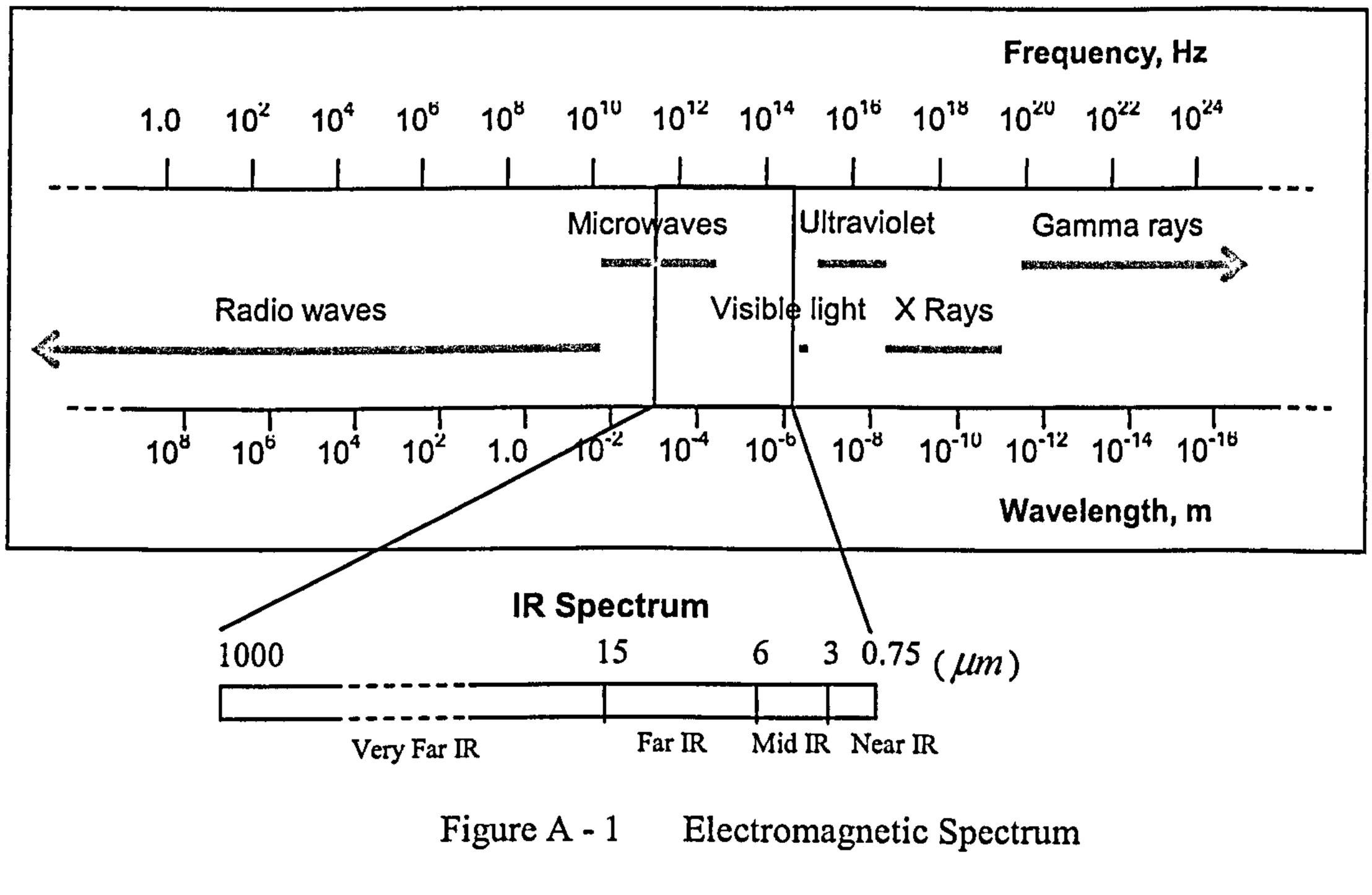


Figure A - 1 Electromagnetic Spectrum

Blackbody Radiation

We start the study of radiation from an ideal body, blackbody, which is characterized by complete absorption of all incident radiation (hence the term black), and maximum possible emission in all wavelength in all directions. In other words, it is the perfect absorber and emitter of all radiation and does not reflect at all. Blackbodies are often used as reference panels to calibrate infrared sensors and images that are taken with spectral band filters. In nature, true blackbody does not exist. However, many objects approximate blackbodies, for example, carbon in its graphite form absorbs all but about 3%. The concept of blackbody emittance is the fundamental of infrared radiation theory.

Stefan Boltzmann found that the irradiance of a blackbody was related to temperature by the law

$$E_{bb} = \sigma T^4 \qquad \text{(A - 1)}$$

Where E_{bb} is the total irradiant (measured by Wm^{-2}) exitance from the surface of a material, σ is the Stefan-Boltzmann constant, $5.6697\times10^{-8} Wm^{-2}K^{-4}$, and T is the absolute temperature (K) of the emitting material.

Equation (A - 1) shows that the magnitude of the total radiant exitance for a blackbody, regardless of directions. It is sometimes necessary to identify the part of the irradiance that is coming from directions within some specified arc of solid angle. The irradiance per unit solid angle is called the radiance L. It can be shown that the irradiance is found by multiplying the radiance with the solid angle:

$$E_{bb} = \int_{\Omega} L \cos\theta \, d\Omega = \pi L \qquad \text{(A - 2)}$$

where θ is the angle between the direction of the radiation and the normal to the surface in question and $d\Omega$ is the solid angle.

Furthermore, Planck showed that the spectral radiant exitance, $E_{bb}(\lambda, T)$, of a blackbody depends on the wavelength λ and the absolute surface temperature T of the object:

$$E_{bb}(\lambda, T) = c_1 / \left[\lambda^5 \left(e^{c_2/\lambda T} - 1\right)\right] \qquad \text{(A - 3)}$$

where $c_1 = 3.7418 \times 10^8 \left(Wm^{-2}\mu m^4\right)$ and $c_2 = 1.4388 \times 10^4 \left(\mu m\, K\right)$.

Figure A - 2 shows a graphical representation of Equation (A - 3) as a function of wavelength and temperature. It shows that at a certain temperature, the black body does radiate energy at every wavelength. The curve gets infinitely close to the x-axis but never touches it. The curve touches at infinite wavelength. It also shows that at a certain temperature, the black body emits at a peak wavelength, at which most of the radiant energy is emitted. At the temperature the graph shows, e.g., from 200K to 2000 K, the peak appears at the Infrared range. As the temperature increases, the peak wavelength emitted by the black body decreases. It will move from the current infrared range to visible range. At 5000K for example, the peak wavelength will be about 5x10-7m (500nm) which is in the visible light region, in the yellow-green section. Objects at around room temperature emit mainly infra-red radiation.

It should be noted that the Stefan-Boltzmann law is expressed for an energy source that behaves as a blackbody. Actual objects only approach this ideal. We will talk about radiation from real materials in the next section.

Equation (A - 1) indicates that the total radiant exitance from the surface of a blackbody varies as the fourth power of absolute temperature. The remote measurement of radiant exitance E from a surface can therefore be used to infer the temperature T of the surface. In essence, it is this indirect approach to temperature measurement that is used in thermal sensing. Radiant exitance M is measured over a discrete wavelength range and used to find the radiant temperature of the radiating surface.

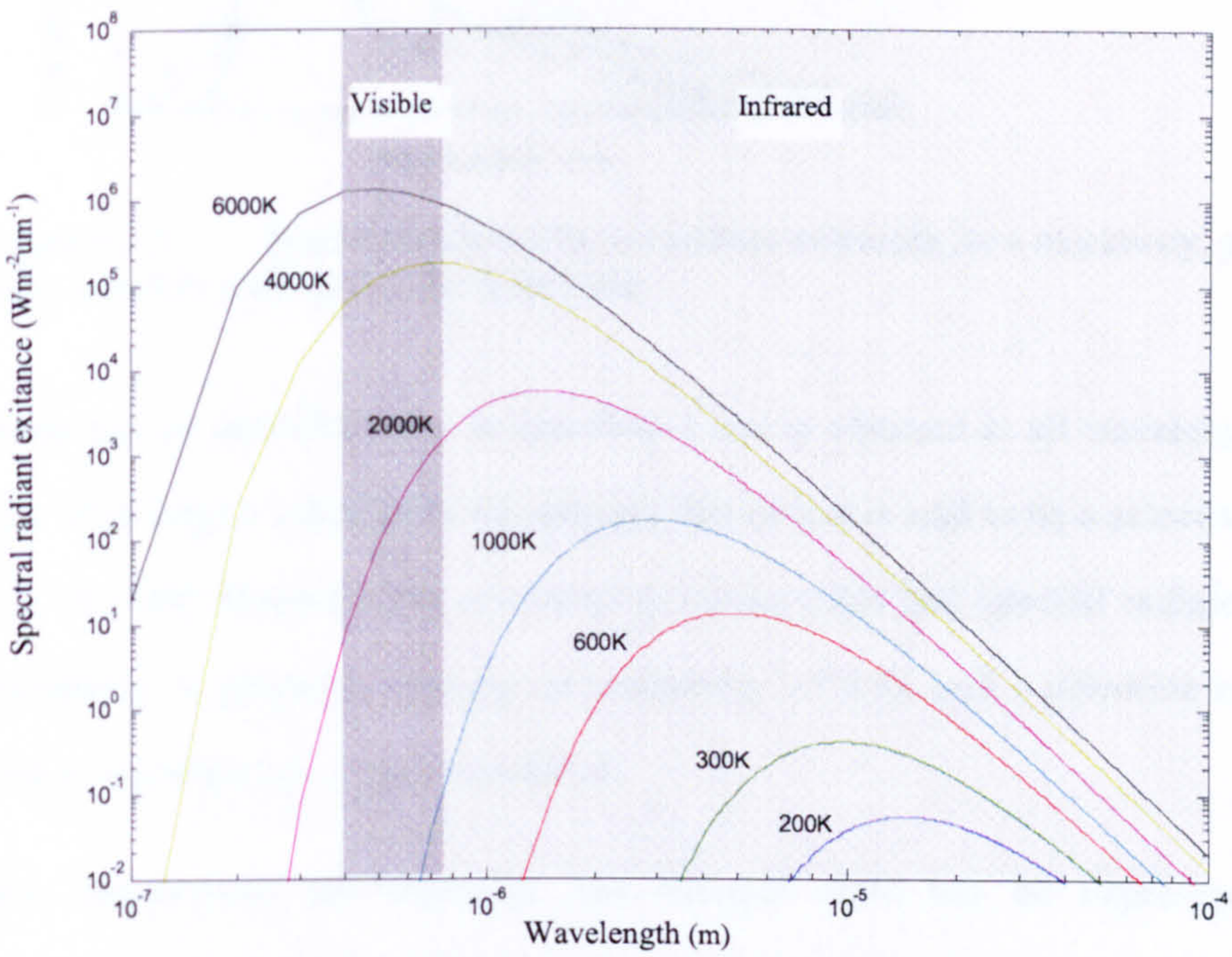


Figure A - 2 Blackbody Radiation Curves

Radiation from real materials

Real materials do not behave as blackbodies. Instead, all real materials emit only a fraction of the energy emitted from a blackbody at the equivalent temperature. The emitting ability of a real material, compared to that of a blackbody, is referred to as a material's emissivity ε. Emissivity ε is a factor that describes how efficiently an object radiates energy compared to a blackbody.

$$\varepsilon = \frac{\text{radiant exitance of an object at a given temperature}}{\text{radiant exitance of a blackbody at the same temperature}} \qquad \text{(A - 4)}$$

As with reflectance, emissivity can vary with wavelength and viewing angle. Depending on the material, emissivity can also vary somewhat with temperature.

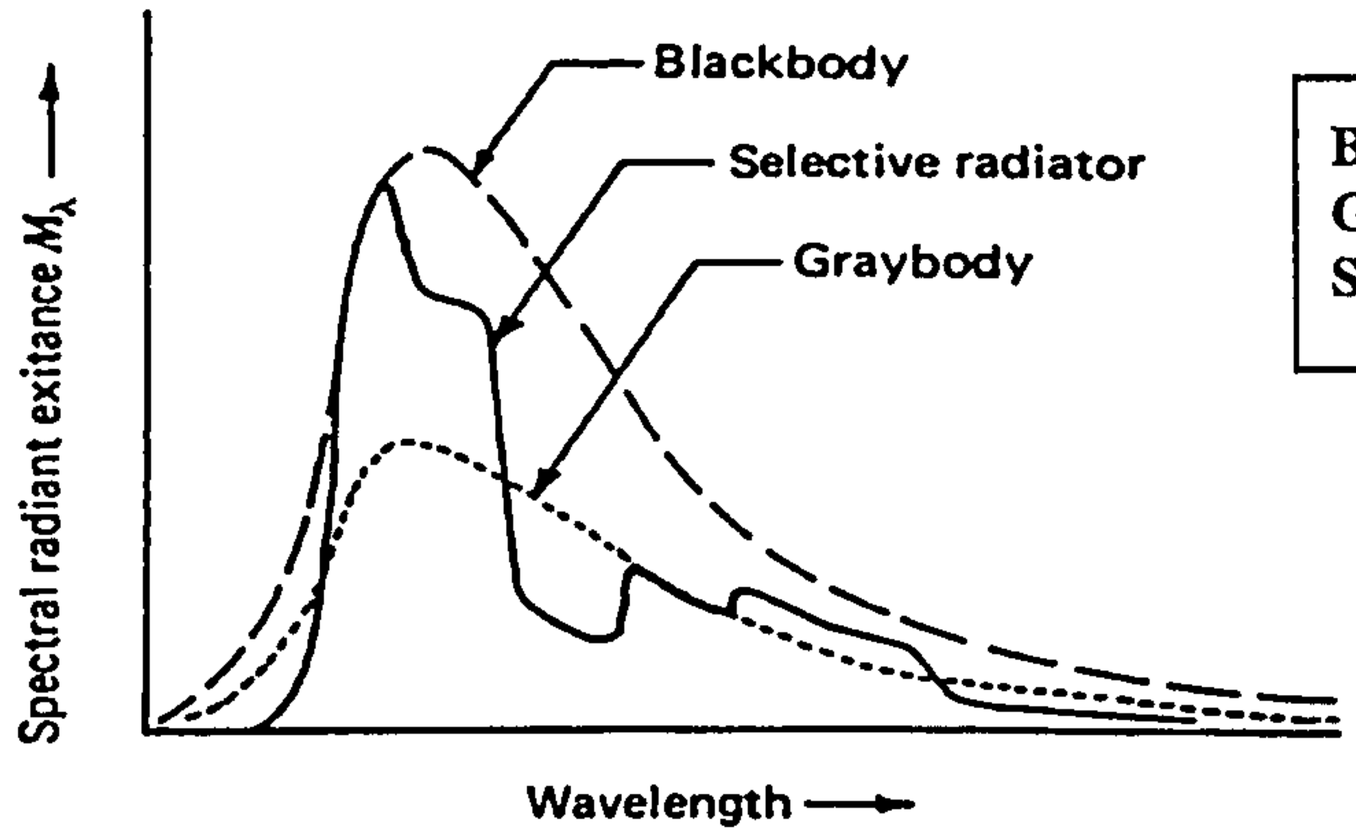


Figure A - 3 Spectral emissivities and radiant exitances for a blackbody, graybody and a selective radiator (source from web)

A graybody has an emissivity that is less than 1 but is constant at all wavelengths. If the emissivity of an object varies with wavelength, the object is said to be a selective radiator. Figure A - 3 [149] illustrates the comparative emissivities and spectral radiate exitances for a blackbody, a graybody (having an emissivity of 0.5), and a selective radiator. In nature, most materials are selective radiator.

A body's temperature can represent one thermal state but be expressed by two temperatures: the first is its internal temperature (from the kinetic motion of its atoms) as measured by an inserted thermometer whereas the second is the external temperature measured by its emitted radiation. As shown in Equation (A - 1), the radiant emanating from a blackbody is related to its internal (kinetic) temperature T. Strictly, this equation holds only for perfect blackbodies; for other bodies (graybodies or selective radiator), the radiance will always be less than the blackbody radiance:

$$E = \varepsilon E_{bb} = \varepsilon \sigma T^4 \qquad \text{(A - 5)}$$

Thus, if we calculate the radiance (sensed) temperature T_r by Equation (A - 1), T_r will be less than the kinetic temperature T:

$$T_r = \varepsilon^{1/4} T \qquad \text{(A - 6)}$$

Figure A - 4 shows that the radiant temperature is significantly higher for a surface with high ε than for a surface with a lower ε, even if the two materials are at the same kinetic temperature.

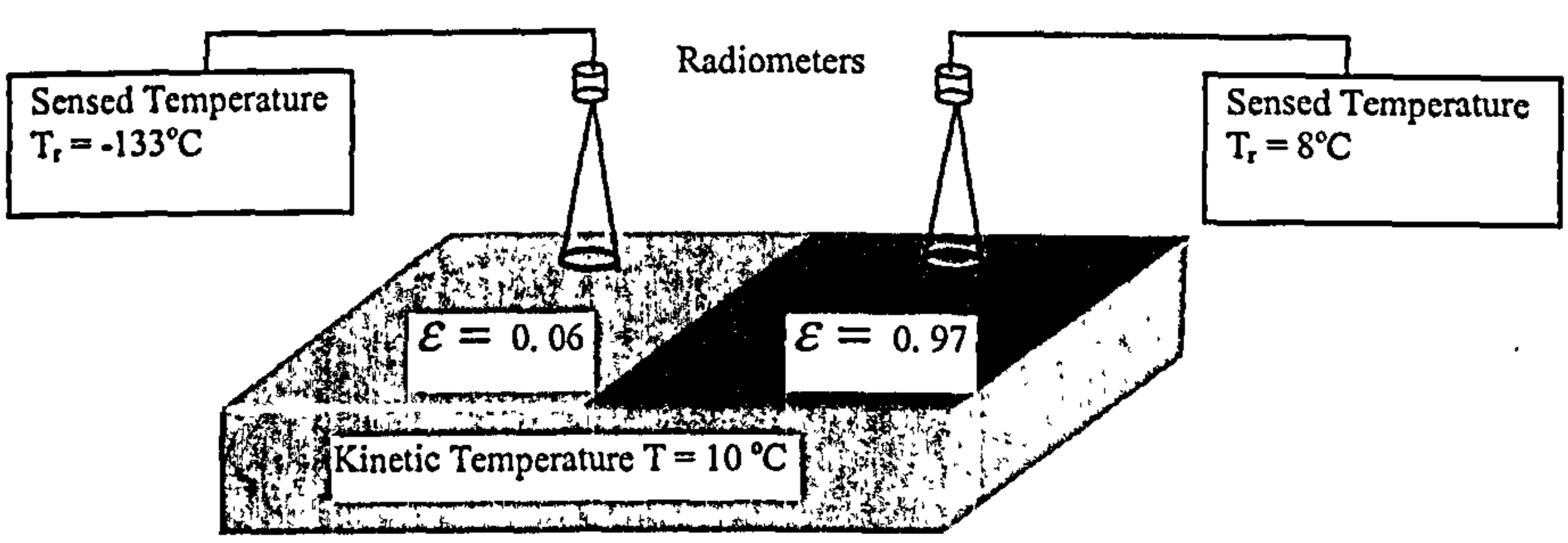


Figure A - 4 Kinetic Temperature and Radiation (Sensed)

Atmospheric Propagation

A simplest target sensor system is shown in Figure A - 5, in which E represents the sum of reflected and emitted radiation from the target[17]. Before this radiation arriving the sensor, the atmosphere absorbs and reflects part of it (E_a and E_r) and only part of it is transmitted E_t. The amount of each part follows the Equation (A - 7):

$$E = E_a + E_r + E_t = aE + rE + tE \qquad \text{(A - 7)}$$

or

$$a + r + t = 1$$

This is simply conservation of energy.

The only reason the sensing devices to detect infrared energy from target is because the atmosphere allows a portion of the infrared energy to be transmitted from the target to the sensor. The wavelength ranges in which the atmosphere is particularly transmissive of energy are referred to as atmospheric windows. Figure A - 6 shows the atmospheric windows and absorption bands. The most efficient absorbers of solar radiation in this regard are water vapour, carbon dioxide, and ozone. For example, atmospheric water vapour absorbs most of the energy in the region from 5-7 um making this region almost

[17] The proportion of each part will depends on materials, surface conditions and temperature, ect.. But in general, in the near earth temperature, in MWIR both reflection and emission are of the same order of magnitude, while in LWIR the emission dominates.

useless for remote sensing. Because these gases tend to absorb electromagnetic energy in specific wavelength bands, they strongly influence "where we look" spectrally with any given remote sensing system. We can see that the wavebands 3-5um and 8-14um are the principal windows for infrared imaging.

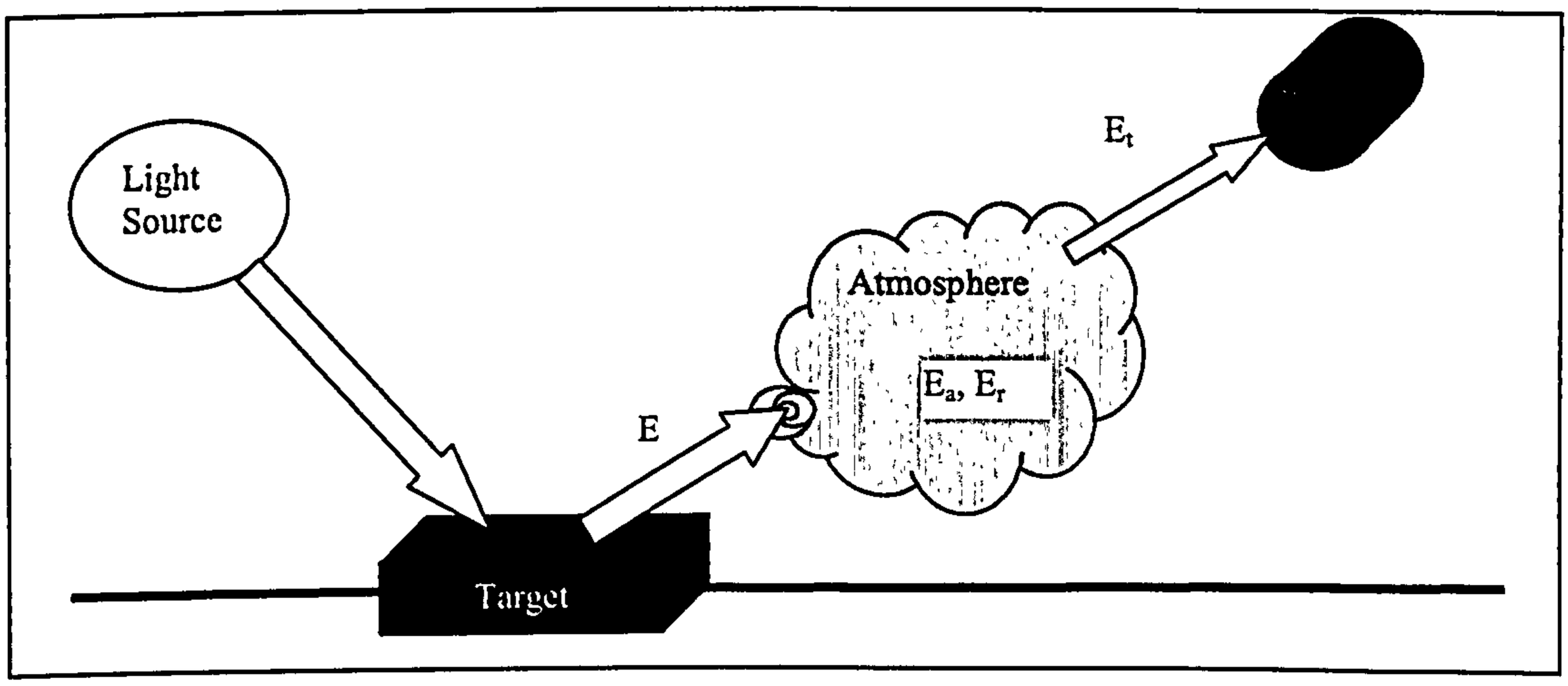


Figure A - 5 A simplest target-sensor system

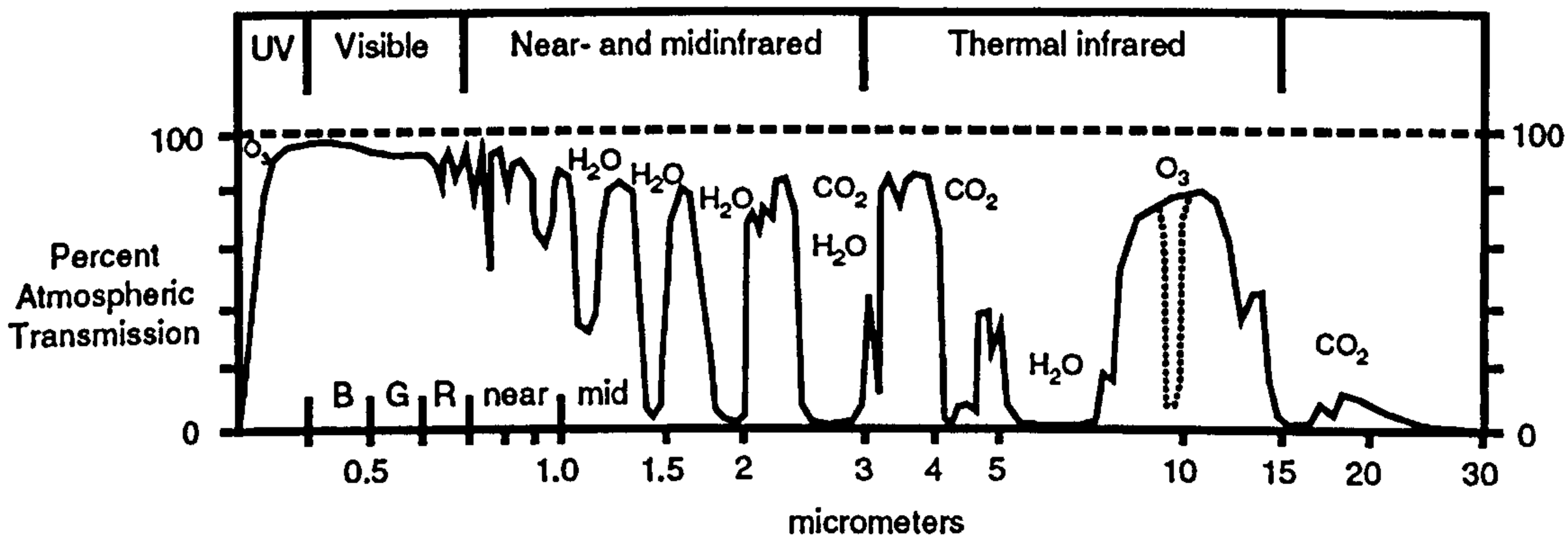


Figure A - 6 Atmospheric Windows and Absorption Bands (Source from web)

The 8um-14um region of spectral radiant exitance is of particular interest since it not only includes an atmospheric window but also contains the peak energy emission for most surface features. The earth's ambient temperature is about 300K. From Wien's displacement law, this means the maximum spectral radiant exitance from earth features occurs at a wavelength of about 9.7 μm. Because this radiation correlates with terrestrial heat, it is themed "thermal infrared" energy. For these reasons, most thermal sensing is

performed in the 8um-14um region of the spectrum. The emissivities of different objects vary greatly with material type in the range. However, for any given material type, emissivity is often considered constant in the 8-14um range when broadband sensors are being used. This means that within this spectral region materials are often treated as graybodies. Figure A - 7 [150] lists typical emissivities of some materials over the range 8-14um.

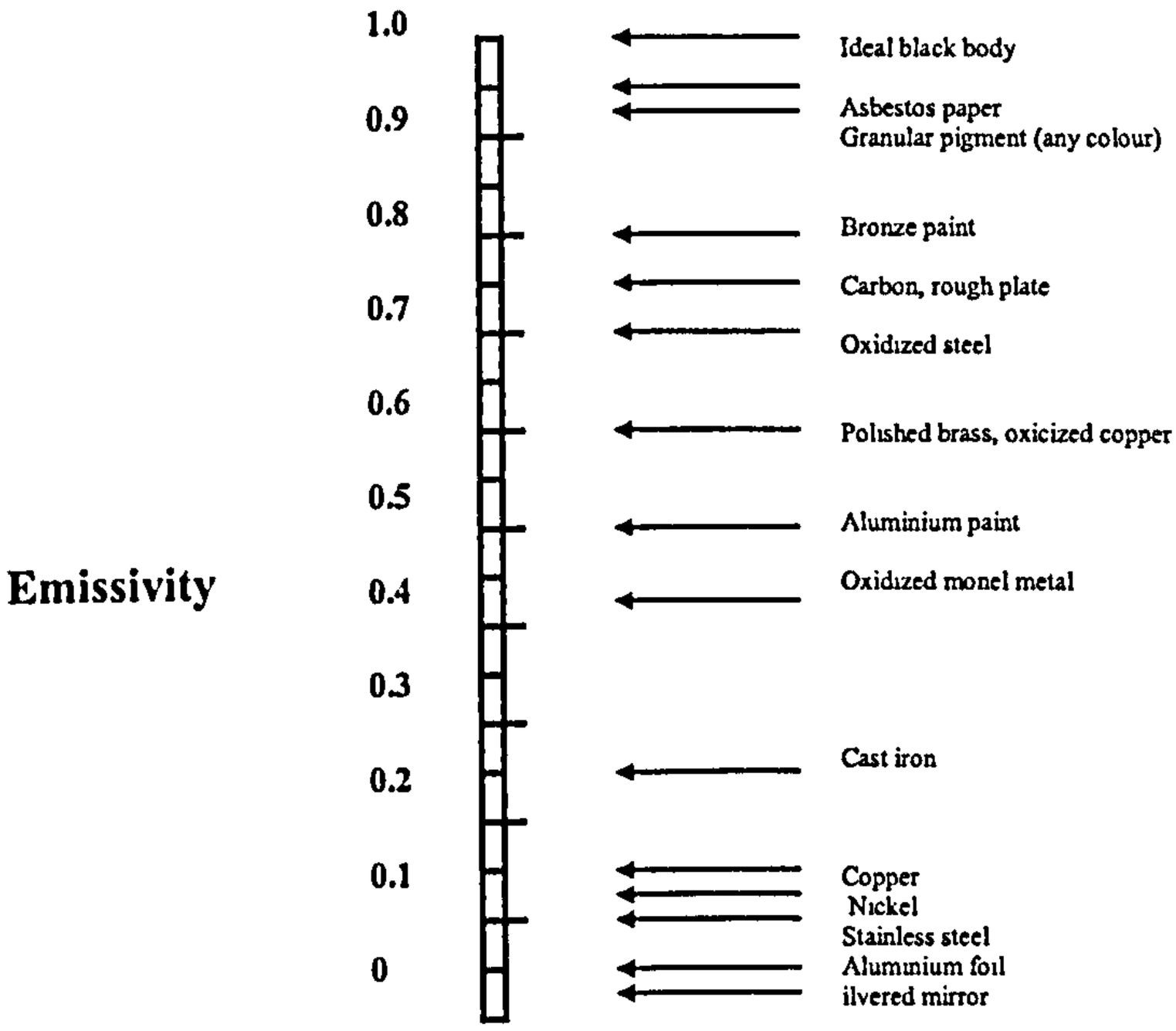


Figure A - 7 Typical Emissivity of Various Common Materials Over the Range of 8 to 14 *μm*

A.2 Infrared Imaging

Because all object above absolute zero emit infrared radiation, the infrared is an excellent spectral region to use for object identification. Using an infrared detector, and object's emitted radiation can be detected and measure.

Infrared Detector

The heart of infrared imaging system is infrared detector. There are two main types of infrared detector: Photon detectors and thermal detectors.

The photon detectors absorb infrared radiation and produce some quantum event, such as the photoelectric emission of electrons from a surface, or electronic interband transitions in semiconductor materials. The output of photon detectors is determined by the rate of absorption of photons and not directly on the photon energy. Example of photon infrared detectors are Mercury-Cadmium-Telluride (HgCdTe) and Indium-Antimonide (InSb). This type of detector normally require cooling down to cryogenic temperature in order to get rid of excessive dark current, but in return they have larger detectivities and small response times.

In contrast to photon detectors, the operation of thermal detectors depends on a two-step process: the absorption of infrared radiation to raise the temperature of the device; and some temperature-dependent parameter such as electrical conductivity changes. The thermal detector can operate at room temperature, but the sensitivity is lower and the response time is longer than for photon detectors.

Infrared Sensor System

An infrared sensor system is a collection of optical elements and electronic hardware connected to a detector. The optical elements reflect and focus incident radiation from an object onto a focal plane, and electronic hardware attached to the focal plane is used to read out the electrical signals generated by each pixel of the focal plane. The focal plane is projected in space by the front optics, creating an imaginary 2D grid of detectors over the target. The size of the projected detector area in the target plane depends on the size of FPA and the range to the target. The radiation incident at each detector depends on the irradiance of the projected detector area on the target, atmospheric attenuation, spectral filter and system optics.

There are two types of infrared sensor systems: mechanical scanning systems and systems based on detector arrays without scanner. A mechanical scanner utilizes one or more moving mirrors to sample the object plane sequentially in a row-wise manner and project there onto the detector. The advantage is that only one single detector is needed. The

drawbacks are that expensive opto-mechanical parts are needed and the detector response time has to be short. Systems based on detector arrays are a rectangular Focal Plane Array continuously looks at a particular area. The advantage is that no moving mechanical parts are needed. The drawback is that the detector array is more complicated to fabricate. It should be mentioned that the mechanical scanning systems are not necessarily use only one detector, detector arrays as well are used for scanning systems.

A.3 Calibration of an infrared data

In an infrared imaging system, there are two types of calibration have to be considered including:

- Calibration of the sensor system – to relate the detector output voltage to the detector incident radiation by a function f, as $\Delta V_{\text{det}} \sim f(\Delta E_{\text{det}})$;

- Calibration of the collected data – to relate the gray level of display or recording device to the detector output voltage, as $G_{image} \sim g(\Delta V_{\text{det}})$.

After the calibration procedure, the radiance or apparent temperature of any surface area on the target in the real world can be retrievable from the collected data.

Assuming the two types of calibration are all linear, we can have

$$L = aG + L_{off} \tag{A - 8}$$

Where G is the gray level of the image, $L = E/\Omega$, is the target radiance and L_{off} is the DC offset (black level) of L. The radiance L to a blackbody can be calculated by

$$L = \int_{\lambda_0} \{\tau_a(\lambda, r)L(\lambda, T_{bb}) + [1 - \tau_a(\lambda, r)]L(\lambda, T_a)\}R(\lambda)d\lambda \tag{A - 9}$$

Where $\tau_a(\lambda, r)$ is the atmospheric transmissivity over the optical path between the imager and target; r is the system-to-target range (m); $R(\lambda)$ is relative system spectral response; T_{bb} is the apparent target temperature (K); and T_a is the air temperature (K). Because Equation (A - 8) has two unknown variables a and L_{off}, two independent measurements are required to solve the equation. Also, blackbodies are needed as targets for the measurements.

The above discussion is to calibrate one detector. If there are detector arrays in the system, each one has to be calibrated. Also, may systems use internal calibration blackbodies that are located behind the front optics, thus, the transmission losses are not measured. In this case, external reference blackbodies must be used.

Appendix B

In the proposed algorithm in Section 5.3.3, there are two parts different from traditional Eigenspace method: step 3 in training stage and step 2 in recognition stage. We will detail how to implement these two steps in the following 2 sections.

B.1 Multidimensional line fitting

In section 5.3.3, we stated that the Eigenspace projections of different thermal states from one pose form a line. We need to do line fitting to find the parameters of the line. Here we consider least squares fitting. We start from 2D line fitting and expand the principle to multidimensional line fitting.

2D line fitting:

2D Line equation:

$$mX + c = Y \qquad (m \text{ is the slope and } c \text{ is the y intercept}) \qquad \text{(B - 1)}$$

For example, if we know 3 points on the line, (1,1), (2,3), (3,7), the line equations are:

$$\begin{aligned} m + c &= 1 \\ 2m + c &= 3 \\ 3m + c &= 7 \end{aligned} \qquad \text{(B - 2)}$$

In matrix form, with the residual vector r, they turn to be

$$\begin{bmatrix} 1 & 1 \\ 2 & 1 \\ 3 & 1 \end{bmatrix} \begin{bmatrix} m \\ c \end{bmatrix} = \begin{bmatrix} 1 \\ 3 \\ 7 \end{bmatrix} - \begin{bmatrix} r_1 \\ r_2 \\ r_3 \end{bmatrix} \qquad \text{(B - 3)}$$

The above equation is an example of the model which has the form of $Ax = d$ where A is an m by n matrix with m larger than n, that is, there are more rows or equations than the unknowns in the x vector. Since there are no exact solutions, we try to find x so that the residual vector in $Ax = d - r$ is as small as possible.

$A^T Ax = A^T d$ is called the normal equation associated with the least squares problem. If $A^T A$ has an inverse, then the solution of the normal equation is also a solution of the least squares problem.

Consider the equation,

$$\begin{bmatrix} 1 & 1 \\ 2 & 1 \\ 3 & 1 \end{bmatrix} \begin{bmatrix} m \\ c \end{bmatrix} = \begin{bmatrix} 1 \\ 3 \\ 7 \end{bmatrix} \qquad \text{(B - 4)}$$

$$\Rightarrow \quad \begin{bmatrix} 1 & 2 & 3 \\ 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} 1 & 1 \\ 2 & 1 \\ 3 & 1 \end{bmatrix} \begin{bmatrix} m \\ c \end{bmatrix} = \begin{bmatrix} 1 & 2 & 3 \\ 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 3 \\ 7 \end{bmatrix}$$

$$\Rightarrow \quad \begin{bmatrix} m \\ c \end{bmatrix} = \begin{bmatrix} 3 \\ -7/3 \end{bmatrix}$$

So line $Y = 3X - 7/3$ is closest to the three data points (1,1), (2,3), and (3,7).

3D line fitting

The 3D line equation

$$\frac{X - c_1'}{m_1'} = \frac{Y - c_2'}{m_2'} = \frac{Z - c_3'}{m_3'} \qquad \text{(B - 5)}$$

We force $c_1' = 0$ and $m_1' = 1$, then the equation becomes

$$\frac{X - 0}{1} = \frac{Y - c_2}{m_2} = \frac{Z - c_3}{m_3} \qquad \text{(B - 6)}$$

where $c_n = c_n' - c_1' m_n' / m_1'$ and $m_n = m_n' / m_1'$. The equation above can be expressed by two independent 2D equations:

$$\begin{aligned} m_2 X + c_2 &= Y \\ m_3 X + c_3 &= Z \end{aligned} \qquad \text{(B - 7)}$$

The solution of fitting 3D line is fitting the two independent 2D line equation on the bases of a selected dimension, e.g., X dimension. The choice of the base dimension will affect the fitting result. In practice, I choose the most 'reliable' dimension, the first dimension of Eigenspace.

Multi-Dimensional line fitting

The Multi-Dimensional line equation

$$\frac{D_1 - 0}{1} = \frac{D_2 - c_2}{m_2} = \frac{D_3 - c_3}{m_3} = \cdots = \frac{D_n - c_n}{m_n} \tag{B - 8}$$

The equation above can be expressed by $(n-1)$ 2D equations:

$$\begin{aligned} m_2 D_1 + c_2 &= D_2 \\ m_3 D_1 + c_3 &= D_3 \\ &\vdots \\ m_n D_1 + c_n &= D_n \end{aligned} \tag{B - 9}$$

Similar as 3D, the MD line can be fitted by fitting (n-1) independent 2D lines.

B.2 Calculate the distance between a point and a line in multidimensional space

The second step in the proposed recognition stage need to calculate the distance between a point and a line in multidimensional space. This section will give details of the calculation.

Let's consider the distance $d(P,L)$ from an point P and a line L (see Figure). Since we are going to use vectors to represent both the point and the line, the dimensionality does not matter. The line L is given by a parametric equation:

$$P(t) = P0 + t(P1 - P0) \tag{B - 10}$$

And we suppose Pb is the base of the perpendicular dropped from P to L. The vector $P0\,P(b)$ is the projection of vector $w = P0\,P$ on to the vector $v = P0\,P1$. We get that:

$$b = \frac{d(P0, P(b))}{d(P0, P1)} = \frac{|w|\cos\theta}{|v|} = \frac{w \cdot v}{|v|^2} = \frac{w \cdot v}{v \cdot v} \tag{B - 11}$$

From the two equations above, we can get the position of $P(b)$:

$$P(b) = P0 + b * v = P0 + \frac{w \cdot v}{v \cdot v} * v \tag{B - 12}$$

The distance between the point and the line $d(P,L) = |P - P(b)|$ can then be easily computed. This computation can also work for any dimensionality.

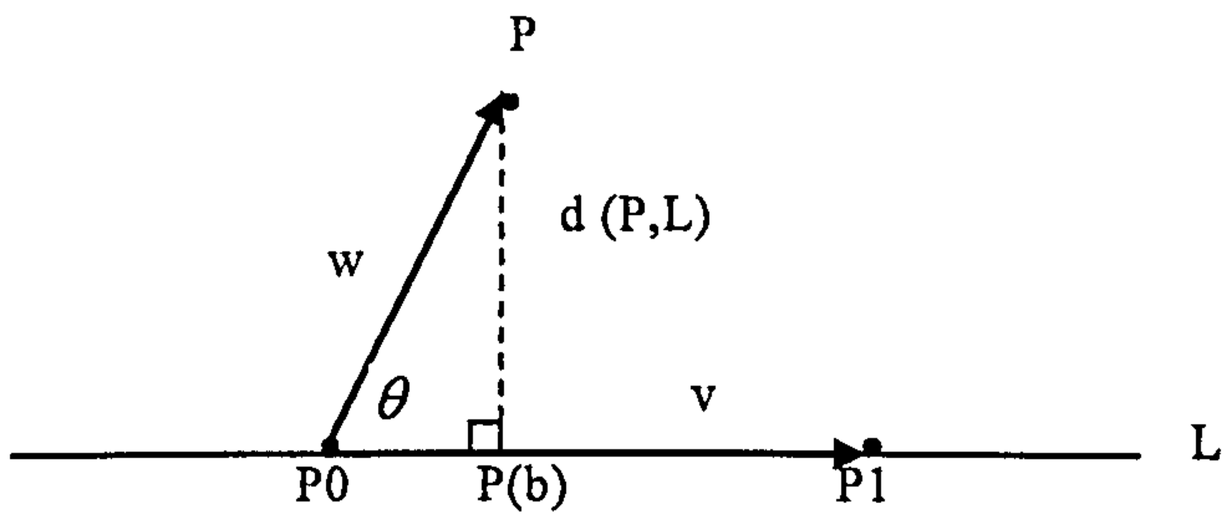


Figure B-1 Distance between a point and a line

Calculate distance between a point and a line in Multi-dimensional space (Pseudo Code)	
1. v=P1-P0	% generate vector v
2. w=P-P0	% generate vector w
3. c1=w • v	
4. c2=v • v	
5. b=c1/c2	% calculate b using Equation (B - 11)
6. Pb=P0+b*v	% calculate point Pb using Equation (B - 12)
Return the distance between P and Pb	

Reference:

1 S. Der, A. Chan, N. Nasrabadi and H. Kwon, Automated vehicle detection in forward-looking infrared imagery, *Applied Optics*, 43(2): 333-348, 2004

2 D. Klick, P.Blumenau and J. Theriault, Detection of targets in infrared clutter, *Proceedings of SPIE*, 4370: 120-133, 2001

3 L. Andreone, P.C. Antonello, M. Bertozzi, Vehicle detection and localization in Infra-red images, *The IEEE 5th International Conference on Intelligent Transportation Systems*, Singapore, 3-6 September 2002

4 S. Sun and H.W. Park, Automatic target recognition using boundary partitioning and invariant features in forward-looking infrared images, *Opt. Eng.*, 42(2): 524-533, 2003

5 J.A. Ratches, C.P. Walters, R.G. Buser, and B.D. Guenther, Aided and automatic target recognition based upon sensory inputs from image forming systems, *IEEE Trans. Pattern Analy. Machine Intell.*, 19(9): 1004-1019, 1997

6 K. Augustyn, A new approach to automatic target recognition, *IEEE Transactions on Aerospace and Electronic Systems*, 28(1), 1992

7 S.Z. Der and R. Chellappa, Probe-based automatic target recognition in infrared imagery, *IEEE Transaction on Image Processing*, 6(1), 1997

8 A. Yilmaz, K. Shafique, M. Shah, Target tracking in airborne forward looking infrared imagery, *Image and Vision Computing*, 21: 623-635, 2003

9 E.T. Lim, C.W. Chan, and V. Ronda, Dim point target detection and tracking system in IR imagery, *Proc. SPIE Visual Communications and Image Processing*, 4067: 277-284, 2000

10 M. A. Zaveri, S. N. Merchant, U. B. Desai, Tracking of point targets in IR image sequence using multiple model based particle filtering and MRF based data association, *17th International Conference on Pattern Recognition (ICPR'04)*. 4: 729-732, 2004

11 R. Chellappa and Q. Zheng, Experimental evaluation of FLIR ATR approaches – a comparative study. *Computer Vision and Image Understanding*. 84:5-24, 2001

12 J. Wilder, P.J. Phillips, C. Jian, and S. Wiener, Comparison of visible and infra-red imagery for face recognition, *Proceedings of the 2nd International Conference on Automatic Face and Gesture Recognition*, 182-187, Killington, VT, 1996

13 D.A. Socolinsky, A. Selinger and J.D. Neuheisel, Face recognition with visible and thermal infrared imagery, *Computer Vision and Image Understanding* 91: 72-114, 2003

14 B. Moghaddam and A. Pentland. Probabilistic visual learning for object representation, *IEEE Trans. on Pattern Analysis and Machine Intell.* 19(7):696-710, 1997

15 A. Leonardis and H. Bischof, Dealing with occlusion in the Eigenspace approach, *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 453-458, San Juan, Puerto Rico, June 1997

16 A. Leonardis and H. Bischof, Robust recognition using Eigenimages, *Computer Vision and Image Understanding*, 78:99-118, 2000

17 R.J. Campbell and P.J. Flynn, A survey of free-form object representation and recognition techniques, Computer *Vision and Image Understanding*, 81:166-210, 2001

18 Y. Lamdan, J.T. Schwartz, and H.J. Wolfson. Affine invariant model-based object recognition. *IEEE trans. Robot. And Automat.*, 6(5): 578-589, 1990

19 K. Arbter, W. E. Snyder, H. Burkardt, and G. Hirzinger. Application of affine-invariant fourier descriptors to recognition of 3-D objects. *IEEE Trans. Pattern Analy. Machine Intell.*, 12:640-647, 1990

20 R. Cyganski and J.A. Orr, Applications of tensor theory to object recognition and orientation determination, *IEEE Trans. Patt. Anal. Mach. Intell.*, 7:662-673, 1985

21 R.L. Cosgriff, Identification of Shape, Ohio State University Research Foundation, Columbus, Rep. 820-11, ASTIA AD 254 792, 1960

22 G. Lei, Recognition of planar objects in 3-D space from single perspective views using cross ratio. *IEEE Trans. Robot. And Automat.*, 6(4):432-437, 1990

23 D. Forsyth, J.L. Mundy, A. Zisserman, C. Coelho, A. Heller, and C. Rothwell, Invariant descriptors for 3D object recognition and pose, *IEEE Trans. Pattern Analy. Machine Intell.*, 13(10):971-991, 1991

24 M.Swain and D.Ballard, Colour indexing, *Int'l J. Computer Vision*, 7(1):11-32, 1991

25 D.Slater and G. Healey, The illumination-invariant recognition of 3D objects using local colour invariants, *IEEE Trans. Pattern Analysis and Machine Intelligence*, 18(2):206-210, 1996

26 A.M. Wallace. A comparison of approaches to high level image interpretation. *Pattern Recognition*, 21(3):241--249, 1988

27 C.A. Rothwell, Object Recognition Through Invariant Indexing, Oxford University Press, 1995

28 E. Trucco and A. Verri, Introductory Techniques for 3-D Computer Vision, Prentic-Hall, Inc. 1998

29 R. Jain, R. Kasturi and B.G. Schunck, Machine Vision, McGraw-Hill, Inc. 1995

30 W.E.L. Grimson, Object Recognition by Computer: The Role of Geometric Constraints, MIT Press, 1990

31 W.E.L. Grimson and T. Lozano-Perez, Localizing overlapping parts by searching the interpretation tree, *IEEE Trans. Pattern Analy. Machine Intell.*, 9(4): 469-482, 1987

32 C.F. Olson, Pose clustering guided by short interpretation trees, *17th International Conference on Pattern Recognition,* 2:149-152, 2004

33 R. Bergevin and M.D. Levine, Generic object recognition: building and matching coarse descriptions from line drawings, *IEEE Trans. Pattern Analy. Machine Intell.*, 15(1):19-36, 1993

34 E.K. Wong, Model matching in robot vision by subgraph isomorphism, *Patt. Recogn.* 25(3):287-304, 1992

35 J. Koenderink, and A. van Doorn, The internal representation of solid shape with respect to vision. *Biological Cybernetics* 32:211-216, 1979

36 J. Koenderink, and A. van Doorn, Surface shape and curvature scales, *Image Vision Comput.* 10:557-565, 1992

37 J. Stewman and K. Bowyer, Creating the perspective projection aspect graph of polyhedral objects. *Int'l Conf. on Computer Vision*, 9:494-500, 1988

38 I. Shimshoni and J. Ponce. Finite-resolution aspect graphs of polyhedral objects. *IEEE Trans. Pattern Analy. Machine Intell.,* 19(4):315-327, 1997

39 D. Eggert and K. Bowyer. Computing the perspective projection aspect graph of solids of revolution, *IEEE Trans. Pattern Analy. Machine Intell.,* 15(2):109-128, 1993

40 D. Kriegman and J. Ponce, Computing exact aspect graphs of curved objects: Solids of revolution, *Int'l Journ. on Computer Vision,* 5(2):119-135, 1990

41 D. Eggert, K. Bowyer, C. Dyer, H. Christenson, and D. Goldgof, The scale space aspect graph, *IEEE Trans. Pattern Analy. Machine Intell.*,15(11):1114-1130, 1993

42 K. Ikeuchi and T. Kanade, Automatic generation of object recognition programs, *Proceedings of the IEEE* , 76(8):1016-1035, 1988

43 S.J. Dickinson, A.P. Pentland, and A. Rosenfeld, From volumes to views: an approach to 3D object recognition, *Computer Vision, Graphics and Image Processing: Image Understanding*, 55(2):130-154, 1992

44 T. Joachims, Text categorization with support vector machines: learning with many relevant features, *Proceedings of ECML-98, 10th European Conference on Machine Learning*, 137-142, 1998

45 S. Tong and D. Koller, Support vector machine active learning with applications to text classification, *Proceedings of ICML-00, 17th International Conference on Machine Learning*, 2000

46 H. Lodhi, J. Shawe-Taylor, N. Christianini and C. Watkins, Text classification using string kernels. *Journal of Machine Learning Research,* 2(3), 2002

47 N. Cristianini, J. Shawe-Taylor and H. Lodhi, Latent Semantic Kernels, *Journal of Intelligent Information Systems*, 18(2), 127-152, 2002

48 T. Leung and J. Malik, Representing and recognizing the visual appearance of materials using three-dimensional textons, *IJCV*, 43(1): 29-44, 2001

49 O.G. Cula and K.J. Dana, Compact representation of bidirectional texture functions. *Proc. Computer Vision and Pattern Recognition*, 1:1041-1047, 2001

50 M.Varma, A. Zisserman, Classifying images of materials: Achieving viewpoint and Illumination independence, *Proc. European Cof. Computer Vision*, 3: 255-271, 2002

51 S. Lazebnik, C. Schmid, J. Ponce, A spars texture representation using local affine regions. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 27(8), 2005

52 G. Csurka, C.R. Dance, L. Fan, J. Willamowski, C. Bray, Visual categorization with bag of keypoints, *In workshop on Statistical Learning in Computer Vision, ECCV*, 1-22, 2004

53 L. Zhu, A. Rao, and A. Zhang, Theory of Keyblock-based image retrieval, *ACM Transactions on Information systems*, 20(2):224-257, 2002

54 J. Vogel and B. Schiele, A semantic typicality measure for natural scene categorization. *In DAGM'04 Annual Pattern Recognition Symposium, Tuebingen, Germany*, 2004

55 K. Mikolajczyk and C. Schmid, An affine invariant interest point detector, *ECCV*(1) 128-142, 2002

56 D.G. Lowe, Object recognition from local scale-invariant features, *Proc. of International Conference of Computer Vision, Corfu*, 1999

57 L. Fei-Fei and P. Perona, A Bayesian hierarchical model for learning natural scene categories, *in CVPR* (2): 524-531, 2005

58 J. Sivic, B. Russell, A.A. Efros, A. Zisserman, and B. Freeman, Discovering objects and their location in images, *International Conference on Computer Vision (ICCV 2005)*, October, 2005

59 V. Vapnik, Statistical Learning Theory, Wiley, 1998

60 T. Hofmann, Unsupervised learning by probabilistic Latent semantic analysis, *Machine learning*, 42: 177-196, 2001

61 D. Blei, A. Ng, and M. Jordan, Latent dirichlet allocation, *Journal of Machine Learning Research*, 3: 993-1022, 2003

62 E. B. Sudderth, A. Torralba, W.T. Freeman, and A.S. Willsky, Learning Hierarchical models of scenes, objects, and parts, *Proc. ICCV*, 11: 1331-1338, 2005

63 R. Alferez and Yuan-Fang Wang, Geometric and illumination invariants for object recognition, *IEEE Trans. Pattern Analy. Machine Intell.*, 21(6):505-539, 1999

64 G. Mamic and M. Bennamoun, Representation and recognition of 3D free-form objects. *Digital Signal Processing*, 12: 47-76, 2002

65 L. Sirovich and M. Kirby, Low-dimensional procedure for the characterization of human faces, *Journal of the Optical Society of America*, 4: 519-524, 1987

66 M. Kirby and L. Sirovich, Application of the Karhunen-loeve procedure for the characterization of human faces, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(1): 103-108, 1990

67 M. Turk and A. Pentland, Eigenfaces for recognition, *Journal of Cognitive Neuroscience*, 3(1): 71-86, 1991

68 H. Murase and S.K. Nayar, Parametric Eigenspace representation for visual learning and recognition, *SPIE Geometric Methods in Computer Vision*, 2031:378-391, 1993

69 H. Murase and S.K. Nayar, Visual learning and recognition of 3-D objects from appearance, *International Journal of Computer Vision*, 14:5-24, 1995

70 C.Y. Huang, O.I. Camps, Object recognition using appearance-based parts and relations, *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 887-883 San Juan, Puerto Rico, June 1997

71 E. Hadjidemetriou and S.K.Nayar, Appearance matching with partial data, *DARPA Image Understanding Workshop (IUW)*, 1071-1078, Nov, 1998.

72 J.Rissanen, A universal prior for the integers and estimation by minimum description length, *Ann. Statist.*, 11(2): 416-431, 1983

73 K. Ohba and K. Ikeuchi, Detectability, uniqueness, and reliability of eigen windows for stable verification of partially occluded objects. *IEEE Transactions on Pattern Analysis and Machine Intelligence*. 19:1043-1048, 1997

74 M. A. Sipe, D. Casasent, Global feature space neural network for active computer vision, *Neutral Computation and Applications*, 7 (3):195-215, 1998

75 H. Borotschnig and L. Paletta, Appearance-based active object recognition. *Image and Vision Computing*, 18:715-727, 2000

76 H. Murase and S. K. Nayar, Illumination planning for object recognition using parametric Eigenspaces. *IEEE Transactions on Pattern Analysis and Machine Intelligence*. 16:1219-1227, 1994

77 A. Fitzgibbon and A. Zisserman, Joint manifold distance: a new approach to appearance based clustering, *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2003

78 P. Simard, Y. Le Cun, J. Denker, and B. Victorri, Transfoamation invariance in pattern recognition-tangent distance and tangent propagation, *in Lecture Notes in Computer Science*, 1524: 239-274, Springer, 1998

79 N. Jojic, B. Frey, P. Simard, and D. Heckerman, Learning mixtures of smoother, nonuniform deformation models for probabilistic image matching, *in Proceedings, AISTATS*, 2001

80 H. Schwenk, and M. Milgram, Constraint tangent distance for online character recognition. *In Proc. ICPR*, D: 515-519, 1996

81 G. Gausorgues, Infrared Thermography, *Microwave Technology*, 5, 1994

82 N. Nandhakumar and J.K. Aggarwal, Integrated analysis of thermal and visual images for scene interpretation, *IEEE Transactions on Pattern Analysis and Machine Intelligence,* 10(4):469-480, 1988

83 N. Nandhakumar, J. Michel, and V. Velten, Thermophysical affine invariants from IR imagery for object recognition, *Proc. of IEEE International Conference on Computer Vision – Physics Based Modelling Workshop*, Bosten, MA, June 1995.

84 J. Michel, N. Nandhakumar and V. Velten, Thermophysical algebraic invariants from infrared imagery for object recognition, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(1):41-51, 1997

85 C.K. Eveland, D.A. Socolinsky and L.B. Wolff, Tracking human faces in infrared video, *Image and vision computing*, 21(7):579-590, 2003

86 A.D. Lanterman, M.I. Miller, D.L.Snyder, Automatic Target recognition via the simulation of infrared scenes, *Proc. of the Sixth Annual Ground Target Modelling and Validation Conference*, 195-204, Keweenaw Research Center, Michigan Tech. Univ., August 1995

87 A.D.Lanterman, M.I.Miller, D.L.Snyder, Representations of thermodynamic variability in the automated understanding of FLIR scenes, *Automatic Object Recognition VI, Proc. SPIE*, 2756: 26-37, Ed: Firooz A. Sadjadi, April 1996

88 M.I. Miller, U. Grenander, J.A. O'Sullivan and D.L. Snyder, Automatic target recognition organized via jump-diffusion algorithms, *IEEE Trans. on Image Processing*, 6(1), 1997

89 D. Nair and J.K. Aggarwal, Robust automatic target recognition in second generation FLIR images, *Proc. 3rd IEEE workshop on Appli. of Computer Vision*, 194-201, 1996

90 D. Nair and J.K. Aggarwal, Bayesian Recognition of targets by parts in second generation forward looking infrared images, *Image and Vision Computing*, 18:849-864, 2000

91 A. Khotanzad and Y.H. Hong, Invariant image recognition by Zernike moments, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(5): 489-497, 1990

92 A. Sluzek, Identification and inspection of 2D objects using new moment-based shape descriptors, *Pattern Recognition Letters*, 166:687-697, 1995

93 S. Haykin, Neural Networks: A Comprehensive Foundation, Prentice-Hall, Englewood Cliffs, NJ, 1989

94 S. Grossberg, Nonlinear neural networks: Principles, mechanisms and architectures, *Neural Networks*, 1:1-61, 1988

95 D.W. Ruck, S.K. Rogers, M. Kabrisky, M.E. Oxley, and B.W. Suter, The multilayer perceptron as an approximation to a Bayes optimal discriminant function, *IEEE Trans. Neural Netw.*, 1(4):296-298, 1990

96 R. A. Horn and C. R. Johnson, Matrix Analysis, Cambridge University Press, 1985

97 R.J.Y. Mcleod and M.L. Baart, Geometry and Interpolation of Curves and Surfaces, Cambridge University Press, 1998

98 F. N. Fritsch and R. E. Carlson, Monotone Piecewise Cubic Interpolation, *SIAM J. Numerical Analysis* 17, 1980, 238-246.

99 D. Kahaner, C. Moler and S. Nash, Numerical Methods and Software, Prentice Hall, 1988.

100 C. De Boor, A Practical Guild to Splines, New York: Springer-Verlag, 1978

101 R.G. Keys, Cubic convolution interpolation for digital image processing, *IEEE Trans. Scoustics, Speech, and Signal Processing*, 29(6), 1981

102 W.S. Russell, Polynomial Interpolation Schemes for Internal Derivative Distributions on Structured Grids, *Applied Numerical Mathematics*, Vol. 17, pp. 129-171, 1995

103 D. F. Watson, Contouring: A guide to the analysis and display of special data, Tarrytown, NY: Pergamon, 1992

104 T. Y. Yang, Finite Element Structureal Analysis, Prentice Hall, 1986

105 C. B. Barber, D.P. Dobkin, and H.T. Huhdanpaa, The Quickhull Algorithm for Convex Hulls, *ACM Transactions on Mathematical Software*, 22(4): 469–483 1996

106 Fauvel, J.; Flood, R.; and Wilson, R. J. (Eds.). Möbius and his Band: Mathematics and Astronomy in Nineteenth-Century Germany. Oxford, England: Oxford University Press, 1993.

107 J.L. Bentley, Multidimensional binary search trees used for associative searching, *Communications of ACM*, 18:509-517, 1975

108 S.A.Nene and S.K.Nayar, Binary search through multiple dimensions, Technical Report CUCS-018-94, Department of Computer Science, Columbia University, New York, 1994

109 S.A. Nene and S.K. Nayar, A Simple Algorithm for Nearest Neighbour Search in High Dimensions, Technical Report, Columbia University, 1995.

110 S. A. Nene, S. K. Nayar and H. Murase, Columbia Object Image Library (COIL-100), Technical Report No. CUCS-006-96

111 K. Ohba and K. Ikeuchi, Detectability, uniqueness, and reliability of eigen windows for stable verification of partially occluded objects, *IEEE Trans. Pattern Anal. Mach. Intell.* 19(9):1043–1047, 1997

112 H. Murase and S. K. Nayar, Image spotting of 3D objects using parametric Eigenspace representation, *The 9th Scandinavian Conference on Image Analysis* (G. Borgefors, Ed.), 1: 323–332, Uppsala, 1995.

113 M. A. Fischler and R. C. Bolles, Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography, *Commu. ACM*, 24(6): 381–395, 1981

114 F. Schaffalitzky and A. Zisserman, Viewpoint invariant texture matching and wide baseline stereo, in *Proc. 8th ICCV*, Canada, 2001

115 T. Tuytelaars and l.V. Gool, Wide baseline stereo matching based on local, affinely invariant regions, in *Proc. 11th BMVC*, 2000

116 A. Leonardis, A. Gupta, and R. Bajcsy, Segmentation of range images as the search for geometric parametric models, *Int. J. Computer Vision* 14(3): 253–277, 1995

117 A. Leonardis and H. Bischof, An efficient MDL-based construction of RBF networks, *Neural Networks*, 11(5): 963–973, 1998

118 R. Moorhead, M.A. Gilmore, A.W. Houlbrook, D.E. Oxford, D. Filbee, C. Stroud, G. Hutchings and A. Kirk, CAMEO-SIM: a physics-based broadband scene simulation tool for assessment of camouflage, concealment, and deception methodologies, *Optoelectronic Engineering,* 40(9):1896-1905, 2001

119 A.W. Haynes, M.A. Gilmore, D.R. Filbee and C. Stroud, Accurate scene modelling using synthetic imagery, *Proceedings of SPIE* 5075: 85-96, 2003

120 M.A. Gilmore, I.R. Moorhead, D.E. Oxford, T.J. Liddicoat, E.R. Filbee, C.A. Stroud, G. Hutchings and A. Kirk, CAMEO-SIM: a broad-band scene generation system that is 'fit for purpose', *Proceedings of SPIE* 3699: 217-228, 1999

121 W.W. Hsieh, Nonlinear principal component analysis by neural networks, *Tellus*, 53A: 599-615, 2001

122 M.Quist, G. Yuna, Distributional scaling: an algorithm for structure-preserving embedding of metric and nonmetric spaces, *Journal of Machine Learning Research,* 5: 399-420, 2004

123 T.J. Hastie and W. Stuetzle, Principal curves, *Journal of the American Statistical Association*, 84(406): 502-516, 1989

124 M.A. Carreira-Perpinan, A review of dimension reduction techniques. Technical Report CS-96-09, Department of Computer Science, University of Sheffield, 1996

125 N. Kambhatla and T.K.Leen, Dimension reduction by local PCA. *Neural Computation*, 9: 1493-1516, 1997

126 J.B.Tenenbaum, V.D.Silva, J.C.Langford, A global geometric framework for nonlinear dimensionality reduction, *Science,* 290: 2319-2323, 2000

127 J.B.Tenenbaum, Mapping a manifold of perceptual observations. *Advances in Neural Information Processing Systems* 10: 682-688, 1998.

128 S.T.Roweis, L.K.Saul, Nonlinear dimensionality reduction by Locally Linear Embedding, *Science,* 290: 2323-2326, 2000

129 K.Saul, S.T.Roweis, An Introduction to Locally Linear Embedding, Technical Report, 2001.

130 Ming –Hsuan Yang, Extended Isomap for pattern classification, *18th national conference on Artificial intelligence*, 224-229, 2002

131 T. F. Cox and M. A. A. Cox, Multidimensional Scaling, Chapman & Hall, London, 1994

132 I. T. Jolliffe, Principal Component Analysis, Springer Verlag, New York, 1986

133 I. Foster, Designing and Building Parallel Programs, Addison-Wesley, 1995

134 S. A. Nene, S. K. Nayar and H. Murase, Columbia Object Image Library (COIL-20), Technical Report CUCS-005-96, February 1996.

135 P.A. Jacobs, Thermal infrared characterization of ground targets and background, SPIE Press, Washington, 2006

136 A. Berk, L. S. Bernstein, and D. C. Robertson, MODTRAN: A moderate resolution model for LOWTRAN 7, *Spectral Sciences*, Burlington, MA, GL-TR-89-0122, 1989.

137 D. Oxford, M. A. Gilmore, I. R. Moorhead, T. J. Liddicoat, D. Filbee, C. R. Stroud, G. R. Hutchings, and A. Kirk, CAMEO-SIM: A physically accurate broadband EO scene generation system for the assessment of air vehicle camouflage schemes, *Proc. Ninth Annual Ground Target Modelling and Validation Conf.*, pp. 204–213, Houghton, MI, 1998.

138 http://www.opticore.com

139 http://www.cagliari.com

140 http://www.thermoanalytics.com

141 http://www.seas.upenn.edu/~timothee/software_ncut/software.html

142 http://en.wikipedia.org/wiki/Coefficient_of_determination

143 J. Shi and J. Malik, Normalized Cuts and Image Segmentation, *IEEE Trans. on Pattern Analysis and Machine Intell.*, 22(8):888-905, 2000

144 A. Strehl and J.K. Aggarwal, Detecting moving objects in airborne forward looking Infra-Red sequences, *Proc. of IEEE on Computer Vision beyond the Visible Spectrum*, 3-12, 1999

145 Thomas M. Lillesand and Ralph W. Kiefer, Remote Sensing and Image Interpretation, John Wiley & Sons, Inc. USA, 2000

146 L.F.Pau and M.Y.E.Nahas, An Introduction to Infrared Image Acquisition and Classification System, Research studies press LTD. 1983

147 J.B.Campbell, Introduction to Remote Sensing, Taylor & Francis, 2002

148 R.Siegel and J.R.Howell, Thermal Radiation Heat transfer, Taylor & Francis, 2002

149 http://www.geog.umd.edu/homepage/courses/372/spr02/lecture1show1.pdf

150 http://www.fas.org/man/dod-101/navy/docs/es310/intro_EO/Intro_EO.htm

151 S.P. Broek, E.J. Bakker, D. Lange and A. Theil, Detection and classification of infrared decoys and small targets in a sea background, Proc. SPIE Targets and Backgrounds VI: Characterization, Visualization, and the Detection Process. 4029: 70-80, 2000

152 Z. Wu and R. Leahy, An optimal graph theoretic approach to data clustering : theory and its application to image segmentation, IEEE Trans. on Pattern Analysis and Machine Intell., 15(11): 1101-1113, 1993

153 F.A. Sadjadi, C.S.L. Chun, Automatic detection of small object from their infrared state-of-polarization vectors, *Optics Letters*, 28(7), 2003